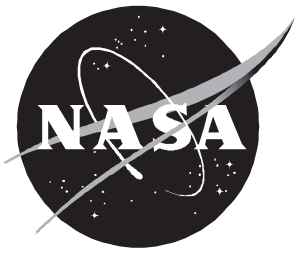
NASA / TM-1998-207640



Aeroacoustic Codes for Rotor Harmonic and BVI Noise - CAMRAD.Mod1 / HIRES: Methodology and Users' Manual

D. Douglas Boyd, Jr.
Virginia Polytechnic Institute and State University, Blacksburg, Virginia

Thomas F. Brooks and Casey L. Burley
Langley Research Center, Hampton, Virginia

J. Ralph Jolly, Jr.
Jolly Development Corporation, Birmingham, Alabama

March 1998

The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the lead center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counter-part or peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that help round out the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results . . . even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at ***http://www.sti.nasa.gov***

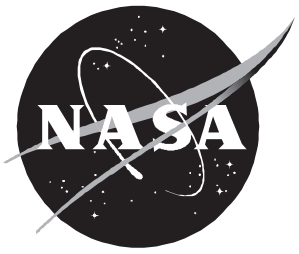
- Email your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA Access Help Desk at (301) 621-0134

- Phone the NASA Access Help Desk at (301) 621-0390

- Write to:
  NASA Access Help Desk
  NASA Center for AeroSpace Information
  800 Elkridge Landing Road
  Linthicum Heights, MD 21090-2934

NASA / TM-1998-207640



Aeroacoustic Codes for Rotor Harmonic and BVI Noise - CAMRAD.Mod1 / HIRES: Methodology and Users' Manual

D. Douglas Boyd, Jr.
Virginia Polytechnic Institute and State University, Blacksburg, Virginia

Thomas F. Brooks and Casey L. Burley
Langley Research Center, Hampton, Virginia

J. Ralph Jolly, Jr.
Jolly Development Corporation, Birmingham, Alabama

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

March 1998

Available from the following:

NASA Center for AeroSpace Information (CASI)
800 Elkridge Landing Road
Linthicum Heights, MD 21090-2934
(301) 621-0390

National Technical Information Service (NTIS)
5285 Port Royal Road
Springfield, VA 22161-2171
(703) 487-4650

Contents

List of Tables

Abstract

This document details the methodology and use of the CAMRAD.Mod1/HIRES codes, which were developed at NASA Langley Research Center for the prediction of helicopter harmonic and Blade-Vortex Interaction (BVI) noise. CAMRAD.Mod1 is a substantially modified version of the performance/trim/wake code CAMRAD. High resolution blade loading is determined in post-processing by HIRES and an associated indicial aerodynamics code. Extensive capabilities of importance to noise prediction accuracy are documented, including a multi-core tip vortex roll-up wake model, higher harmonic and individual blade control, tunnel and fuselage correction input, diagnostic blade motion input, and interfaces for acoustic and CFD aerodynamic codes. Modifications and new code capabilities are documented with examples. A user's job preparation guide and listings of variables and namelists are given.

Chapter 1

Introduction

1.1 Introduction

With growing noise restrictions being imposed on rotorcraft, a means to accurately and efficiently predict noise generated by a wide variety of rotorcraft configurations is needed. Many of the existing rotorcraft computer codes that are available are intended to calculate only rotorcraft performance quantities. The calculation requirements of the rotor system in a performance analysis often involves only the lowest frequency loading results. For example, in level steady flight, a simple performance analysis might only require knowledge of the mean rotor thrust and drag. To predict such quantities, a high resolution loading calculation on the rotor is not necessary. But, for noise calculations, a detailed, high resolution radial and azimuthal loading solution is needed in order to accurately define events such as Blade-Vortex Interaction (BVI) noise. A computer code system designed to fill this requirement is presented in this documentation. This computer code system uses, as a "base" code, the original 1980 version of the Comprehensive Analytical Model for Rotorcraft Aerodynamics and Dynamics (CAMRAD) (Ref. [1] and [2]). This document is intended to supplement the original documentation of the CAMRAD code, not to replace it. It is assumed that the reader is already familiar with the original version of CAMRAD. Since this document enumerates the changes that have been made to the original CAMRAD code to create the code system that is now collectively known as "CAMRAD.Mod1/HIRES" (Ref. [3]), it is intended to document the modifications and to be a reference for the new coding. Presented are the three major parts of the code: CAMRAD.Mod1, HIRES,

and the Indicial Post-Processor. It also contains an updated Users' Manual that lists all variable inputs to the code system.

1.2 Organization of Documentation

This chapter provides an introduction, a system overview, an outline of the documentation, and a discussion of the sample cases to be used in this document. Chapter 2 deals with the modifications made to the low resolution part of the original version of CAMRAD to obtain CAMRAD.Mod1. Within each chapter, a section discusses each set of modifications. Where applicable, the first subsection of each section is an introduction to discuss the motivation for the modification and the methods used. Subsequent subsections discuss details of the modification including the actual code changes. When applicable, the last subsection of each section discusses how the modification is related to the high resolution modifications.

Chapter 3 discusses the changes made to include a high resolution post-processor, known as HIRES. This coding is part of the CAMRAD.Mod1 code, but is executed after the trim loop of the low resolution calculations.

Chapter 4 deals with the code known as the Indicial Post-Processor (IPP). This code is a standalone code that incorporates many aspects of the works of T.S. Beddoes and Gordon Leishmann with regard to empirical use of indicial aerodynamic functions. Many of their formulations are directly applicable in the code, but others were modified such that they could be cast into a form compatible with the CAMRAD.Mod1 system.

Chapter 5 is the Users' Manual and describes namelist inputs, information on codes needed to prepare input data, and other general user-related information for CAMRAD.Mod1. This chapter is intended to be a supplement to Reference [2].

1.3 System Overview

A code system has been developed to expand the capabilities of previous rotorcraft performance and noise codes. As a "base" code, the original version of CAMRAD was chosen. The original CAMRAD version is capable of performing comprehensive rotorcraft calculations such as performance and low resolution loading calculations, for various rotorcraft configurations including a single rotor in a wind tunnel, a conventional helicopter, a tandem rotorcraft, a coaxial rotorcraft, and a tiltrotor. The original analysis

is divided into a several parts. First, a "Trim" analysis determines the rotorcraft configuration (*i.e.,* orientation, control settings, *etc.*) required to match a specified flight condition. Second, a "Flutter" analysis linearizes the rotorcraft equations of motion about the trimmed configuration and determines eigenvalues, *etc.* Third, the "Transient" analysis determines a rotorcraft non-equilibrium response to a particular input such as a gust. The majority of the work in this document is related to the Trim analysis and post-processing of the Trim analysis results. Since the focus of the work in the CAMRAD.Mod1 effort is on the Trim solution and post-processing of the Trim results, no effort has been made to update the Flutter and Transient analyses; as such, their usage in CAMRAD.Mod1 is neither recommended nor are they discussed further in this document.

Since most of the following document pertains to the Trim analysis and new follow-on procedures, a brief review of the trim process used in CAMRAD is in order. It is assumed here that the reader is somewhat familiar with the CAMRAD prediction capability and the details of the rotor trim methods used therein. In Figure 1.1, the box labeled CAMRAD.Mod1, that includes the Trim, Transient, and Flutter analyses, represents the low resolution portion of CAMRAD.Mod1. This set of analyses, discussed above, are analogous to the original CAMRAD code. Also shown are several input paths and an additional output path into HIRES. These paths will be discussed later. Figure 1.2 is an expansion of the Trim box of Figure 1.1. As the trim procedure is being carried out, all processes except the current process are held constant, as in the original CAMRAD code.

For example, first, the wake and the wake influence coefficients are determined for a fixed configuration. That is, the influence coefficients are determined for fixed blade motion, fixed circulation, fixed blade controls, *etc.* Once these wake influence coefficients are known, they are held fixed and the next stage proceeds. After the wake influence coefficients are known, the necessary control settings are determined to match the target flight condition. A modified Newton-Raphson technique is used to increment the controls to determine a guess at the actual control settings required to meet the target flight condition. Then, with the wake influence coefficients fixed, the blade controls fixed, and blade motion fixed, the circulation distribution is determined. With the just calculated circulation distribution, the blade motion is recalculated. This motion/circulation iteration continues until successive iteration differences for both have converged to below a prescribed tolerance. With the motion/circulation iterations converged, the next guess at a control setting is made, and the process is continued until all

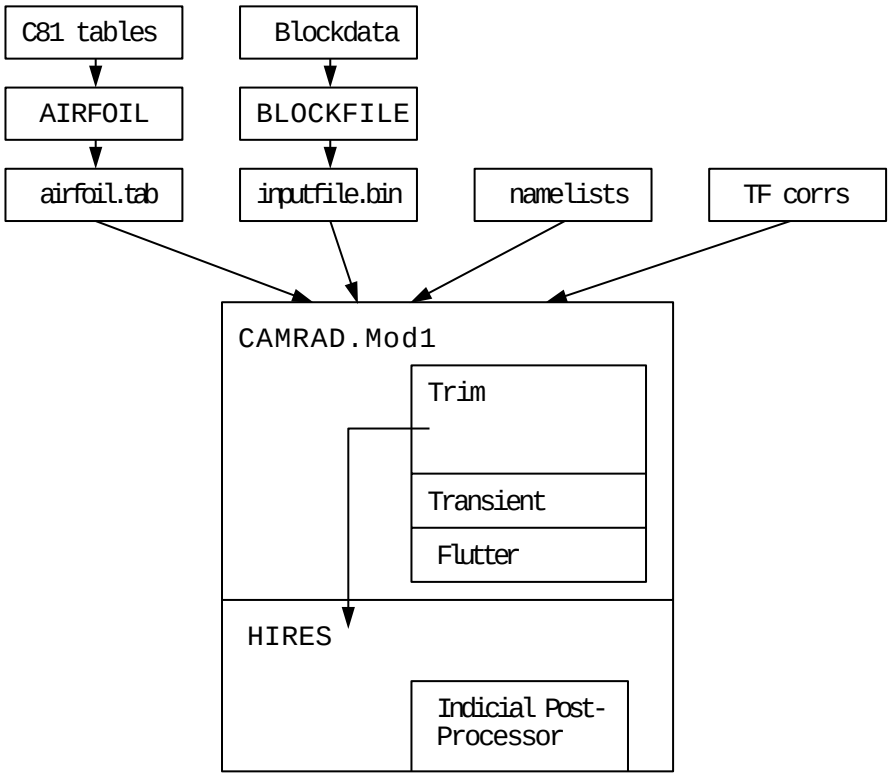


Figure 1.1: Flowchart of CAMRAD.Mod1 and HIRES with associated inputs.

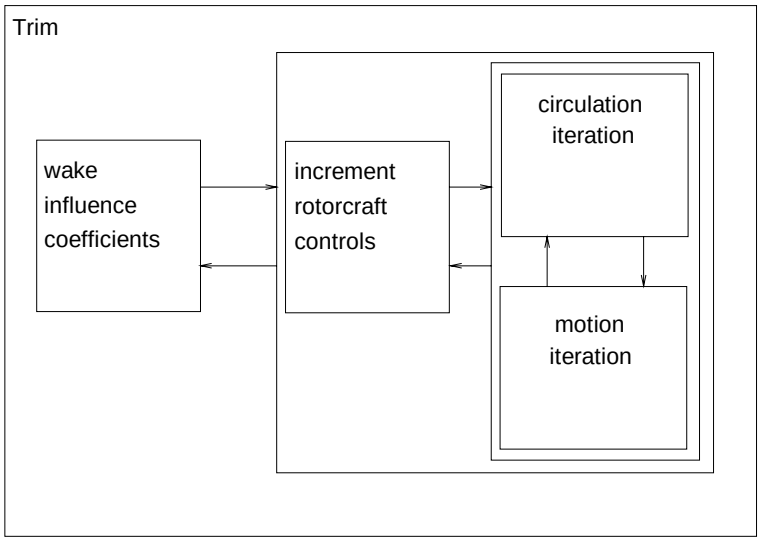


Figure 1.2: Trim loop and lnner loops of CAMRAD.Mod1.

iterations are converged and the target flight condition is met. This entire trim process may be successively repeated for several (or the same) wake models: a uniform inflow model, prescribed wake model, free wake model, and rollup wake model.

Figure 1.1 also illustrates a flow chart of the input requirements for the CAMRAD.Mod1 code system. First, airfoil characteristics (*i.e.,* lift, drag, and moment coefficients) are normally available in a standard "C81 airfoil table". These airfoil tables are used by the airfoil preparation program, hereafter denoted as AIRFOIL, to generate a binary airfoil file, represented by "airfoil.tab" in the figure. Alternatively, airfoil characteristics may be generated using namelist inputs to AIRFOIL. Second, BLOCKDATA information for the rotorcraft is prepared by the input preparation program, hereafter denoted as BLOCKFILE. This information is also normally converted to a binary input file, labeled "inputfile.bin" in the figure, for use in the analysis. In addition to the BLOCKDATA, namelist inputs are used to set specific run conditions such as RPM, advance ratio, *etc.* These namelists are located in the script file or command file used to run the analysis. Also, other files may be input to the analysis for use in the tunnel/fuselage correction model, denoted "TF corrs" in the figure.

Several new output paths emerging from CAMRAD.Mod1 were introduced to predict such events as BVI and system noise. Figure 1.3 again shows the CAMRAD.Mod1 box (a box containing a trim loop, a transient loop, and a flutter loop), and the HIRES box with their associated outputs. As seen in this figure, several new branches have been made out from the end of the trim loop; none of these branches existed for the original version of CAMRAD. The first of these branches, the Trim-FPRBVI branch, along with the possible return branch FPRBVI-Trim, is called the CFD interface and is discussed later in this document. This branch uses the low resolution wake and blade position information and applies an external CFD code, in this case, FPRBVI (Ref. [4]), to calculate the high resolution loading utilized by the rotorcraft noise code, WOPWOP (Ref. [5]), to predict BVI noise. A return path from FPRBVI to the trim loop is possible in an open-loop manner, to take advantage of the loading calculated by the CFD code in the calculation of rotorcraft trim. Along the same branch, once the tone noise has been calculated by WOPWOP, the ROTONET (Ref. [6]) code system could be applied to compute propagated noise.

In the second branch (Trim-ROTONET), the flight condition, the blade position, and the low resolution loading information is made available for use in the systems noise code ROTONET to predict tone noise and propagation

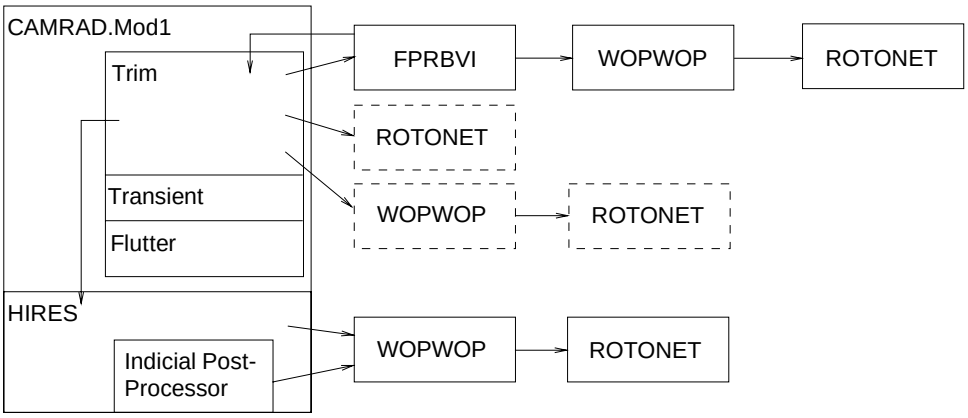


Figure 1.3: CAMRAD.Mod1 and HIRES output paths to other codes.

effects. In the third branch (Trim-WOPWOP), the flight condition, the blade position, and low resolution loading information is made available for use in the rotor tone noise prediction code WOPWOP. The WOPWOP results could then be used in ROTONET to calculate propagated noise. Both of these branches are discussed later in the document. Note in the figure that both of these paths are surrounded by dashed lines. This is done to indicate that, even though these paths exist in the code, their use is not recommended since they do not include higher harmonic loads and their use may produce misleading results.

The next branch involves the extensions of CAMRAD to include a high resolution wake and/or loading calculation known as HIRES. Figure 1.4 provides a brief introduction to the solution procedures used in this portion of the code; a more detailed discussion is provided later in the document. After the trim solution has been obtained, the far wake influence coefficients are obtained using a high resolution reconstruction of the blade position and wake position. With the far wake influence coefficients known, airloads due to the far wake effects may be calculated. To account for the near wake effects, two choices are possible. One choice is a near wake lattice model used to calculate influence coefficients of a near wake lattice, followed by an airload analysis. Although this method exists in the code, is has not been exercised thoroughly or validated. The other choice is an Indicial Post-Processor (IPP) code that accounts for the near wake effects using indicial aerodynamic functions. Once the airloads are known, they may be used in the rotorcraft tone noise code WOPWOP, optionally followed by the ROTONET code to account for noise propagation effects.

1.4 Sample Case

Throughout Chapter 2 of this documentation, a sample case will be used to illustrate modifications to CAMRAD.Mod1. Examples of results from HIRES and the IPP can be found in the literature (Ref. [3]) and thus are not presented here. In general, when comparisons are being made between this sample case and the same case showing the modification, the upper plot of a given figure is the result from the sample case. The lower plot is the same sample case including the particular modification in question. The sample case is a model BO-105 hingless rotor in a wind tunnel. Some of the properties used in the sample case are listed in Table 1.1.

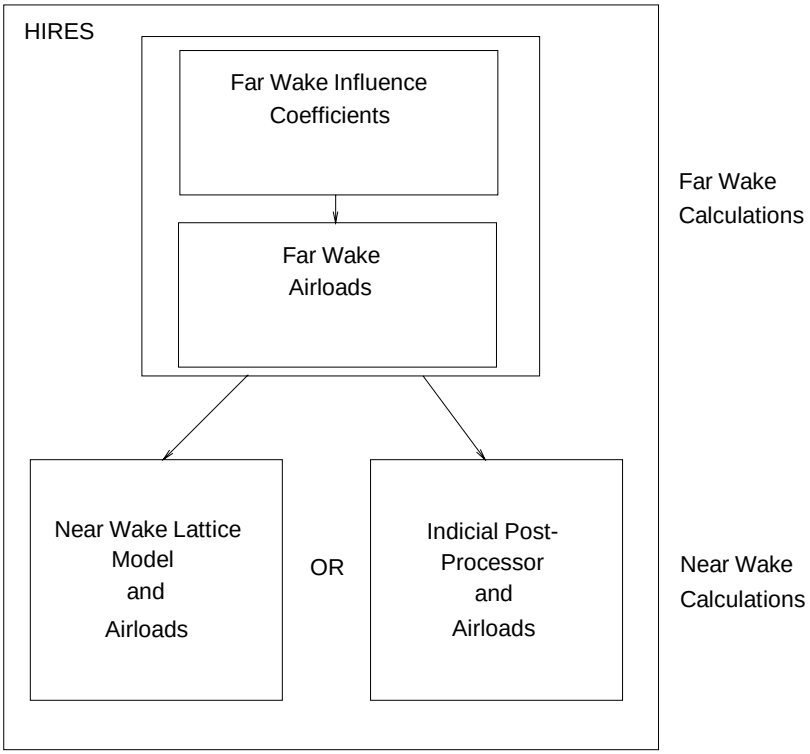


Figure 1.4: The major computational loops within HIRES.

Table 1.1: Sample Case Information

radius	2.0 meters
chord	0.121 meters (rectangular planform)
number of blades	4
flap hinge location	(none)
lag hinge location	(none)
sweep of quarter-chord	0.0 degrees
airfoil section	NACA 23012
precone	0.0 degrees
nominal advance ratio	0.15
nominal RPM	1041.0
nominal shaft tilt	5.3 degrees (aft tilt)
nominal Ct/sigma	0.05607

Chapter 2

CAMRAD.Mod1

In this chapter, modifications made to the low resolution part of CAMRAD are presented. There are a number of sections, each describing in detail the specific modification and/or enhancements made to CAMRAD. Where appropriate original CAMRAD predictions are compared to CAMRAD.Mod1 (modified CAMRAD) predictions.

2.1 Changes to Free Wake Azimuthal Resolution

2.1.1 Introduction

There are two vortex wake models in CAMRAD to determine vortex geometry. These models are the rigid (or prescribed) wake model and the free wake model. One of these wake models is used during the wake influence coefficients calculation (shown in Figure 1.2) to determine the tip vortex geometry. As for the free wake geometry model, CAMRAD.Mod1 relies on the Scully Free Wake method (Ref. [7]) as does the original version of CAMRAD. However, for CAMRAD.Mod1, a higher resolution free wake analysis is desired and the smallest possible azimuth step size was changed from 15° to 10°. Changes were made mostly by redimensioning arrays to allow 10° azimuth steps in the free wake geometry calculations. In the early development of CAMRAD.Mod1, under certain circumstances, the free wake geometry calculations gave an error messages indicating too many "transition points", which caused the program to stop. (For details on transition points, see Ref. [7]). In order to allow CAMRAD.Mod1 a better chance of completing the free wake portion of the program without stopping, the number of allowable transition points was doubled from 16 to 32. One of the

input parameters that the free wake geometry calculation uses from other parts of CAMRAD.Mod1 is the maximum bound circulation. Modifications were made to include choices of different circulation options to be used in the free wake geometry analysis. The options available are as follows: the original maximum bound circulation, the maximum positive bound circulation, the maximum negative bound circulation, the maximum outboard bound circulation, the maximum inboard bound circulation, and the "large core" circulation (discussed in the Section 2.17). The choice of option is dependent on the problem being explored.

2.1.2 Sample Case Discussion

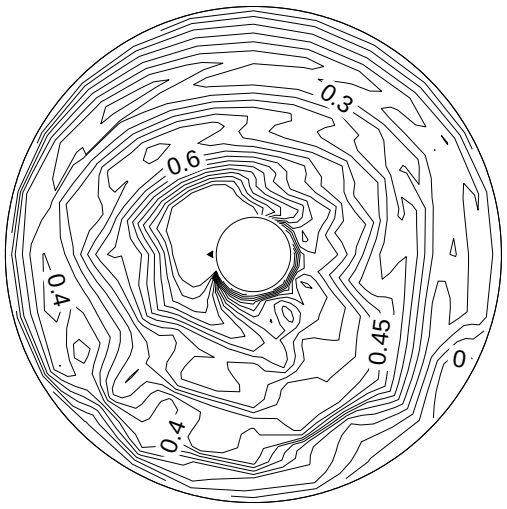
The effect of changing the azimuthal resolution from 15 ° to 10 ° for the sample rotor in a descent condition is shown in Figures 2.1 (a) and 2.1 (b). In this figure, contours of local lift coefficient, C_l, are shown over the rotor disk. These predictions were made using the original maximum bound circulation option in the free wake model. Figures 2.2 (a) and 2.2 (b) show the lift coefficient as a function of span for the 15 ° and 10 ° azimuth step cases for several azimuth locations. The major difference in this case compared to original CAMRAD is the azimuth resolution used for the free wake analysis. Some small differences are seen; but no systematic study of these small differences has been made. However, the primary purpose of the modification is to provide higher resolution wake and blade geometry to HIRES reconstruction than would be possible with the original CAMRAD free wake resolution.

2.1.3 Code Modifications

The following changes were made in the free wake subroutines:

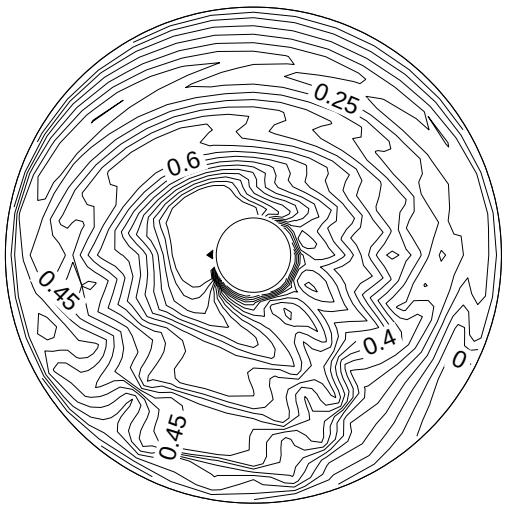
1. Array dimensions were changed as follows in common blocks SQCAL, SSPLOT, and SGAM:

```
(97)          becomes  (145)
(3,25)        becomes  (3,37)
(6,25)        becomes  (6,37)
(3,25,97)     becomes  (3,37,145)
(25)          becomes  (37)
(6,25,16)     becomes  (6,37,16)
```



$\psi = 0.0^\circ$

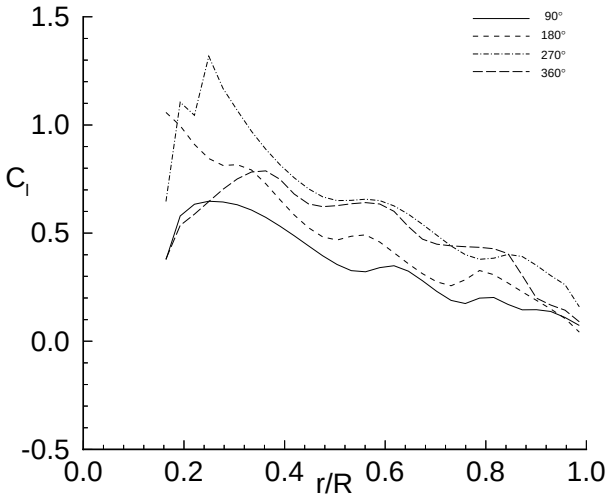
(a) C_l for 15° wake resolution



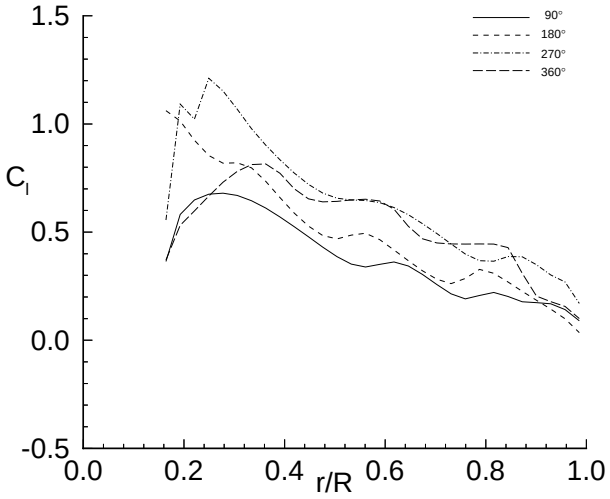
$\psi = 0.0^\circ$

(b) C_l for 10° wake resolution

Figure 2.1: Contours of local lift coefficient C_l over the rotor disk showing effect of higher azimuthal resolution in the free wake model.



(a) C_l for 15 $^\circ$ wake resolution



(b) C_l for 10 $^\circ$ wake resolution

Figure 2.2: The local lift coefficient as a function of span showing the effect of higher azimuthal resolution in the free wake model.

2. The array KTR(6,37,16) in the common block SQCAL, already modified by the redimensioning in (1) above, was changed to KTR(6,37,32). This change was made in subroutines DCALC, GEOMF1, GEOMF2, NWCAL, VSCAL, and WQCAL.

3. In subroutine NWCAL, the line:

```
IF (KM(I,J) .GE. 16) GOTO 490
```

   was changed to:

```
IF (KM(I,J) .GE. 32) GOTO 490
```

4. In the subroutines CHEKR1 and CHEKR2, the line:

```
IF (LEVEL .EQ. 2) .AND. (MPSI .GT. 24) GOTO 21
```

   was changed to:

```
IF (LEVEL .EQ. 2) .AND. (MPSI .GT. 36) GOTO 21
```

5. A change was made in QCVL to handle instances when the variables AL and/or BL are ≤ 0. The following lines were added:

```
IF ((AL .GT. 0.) .AND. (BL .GT. 0.)) THEN
Q = ...          (original line from code)
ELSE
Q = 0.
ENDIF
```

6. Two FORMAT statements were modified to allow proper output of parameters during use of the DEBUG variable. FORMAT statement number 2 was changed such that the variable, H, is output with the "I3" instead of the "I2" format, and the following "3X" was changed to "2X" to retain the same field width. Also, FORMAT statement number 4 was modified so that the "I3" format is used instead of the "I2" format. These FORMAT changes were made in both GEOMF1 and GEOMF2.

7. Options for different maximum circulations used in the free wake calculations are controlled by input parameters OPMXFWG, OPROLLU, and IFWLGC in namelist NLTRIM (see Chapter 5). These parameters were added at the location where, originally, the maximum bound circulation was stored in an array for use in the free wake geometry calculations.

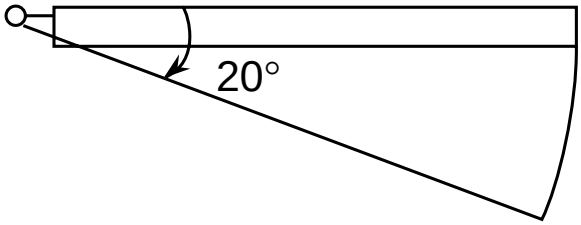
2.1.4 Extensions to High Resolution

The free wake geometry is determined by calculations in the low resolution portion of CAMRAD.Mod1; whereas, in HIRES, the wake geometry is interpolated as needed from the low resolution information. This interpolation is applied between known, low resolution wake endpoints, as discussed in Chapter 3. As such, the modifications discussed here are automatically included in the HIRES reconstruction procedure without further coding considerations.

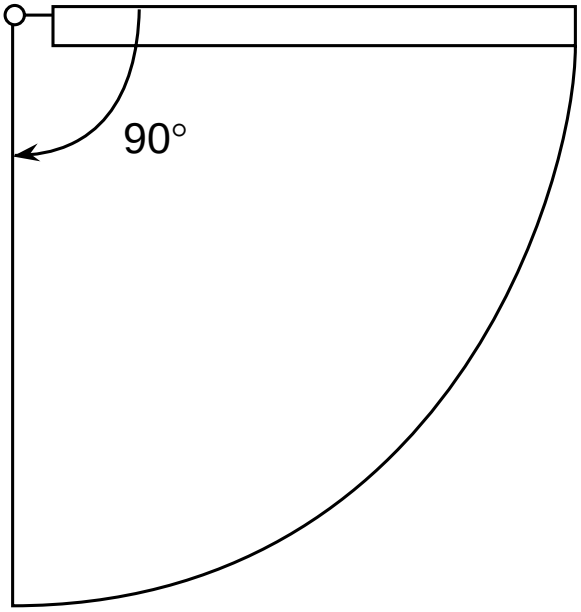
2.2 Modification to Allow 90 Degrees of Near Wake

2.2.1 Introduction

When calculating the near wake portion of the wake influence coefficients in the trim solution (see Figure 1.2), the original version of CAMRAD placed a limit on the extent of wake behind the reference blade that could be designated as near wake. This limit was a function of the number of azimuthal and radial resolution being used and was due only to array sizes in the code. In order to have the capability to test the effects of an extended near wake, a modification was made to the code to allow up to 90 $^\circ$ of near wake. Originally if using the maximum number of radial stations (MRA = 30), combined with 36 azimuth steps (10 $^\circ$ steps), one was limited to 20 $^\circ$ of near wake (KNW =2). Modifications were made so that under these conditions, one could use up to 90 $^\circ$ of near wake (KNW = 9). Figure 2.3 illustrates the modification. The near wake is represented symbolically by a circular arc, whereas in the code, the near wake is a vortex lattice model.



(a) original CAMRAD prediction for a near wake extended 20°,
MRA=30, KNW=2, $\Delta\psi = 10°$



(b) CAMRAD.Mod1 prediction for a near wake extended 90°,
MRA=30, KNW=9, $\Delta\psi = 10°$

Figure 2.3: Extent of the near wake ilustrated for the original CAMRAD and CAMRAD.Mod1.

2.2.2 Sample Case Discussion

Figures 2.4 (a) and 2.4 (b) show the contours of lift coefficient using 20 ° of near wake and 90 ° of near wake, respectively. Figures 2.5 (a) and 2.5 (b) show the same information plotted at a several azimuth stations. Though there are not large effects apparent in these plots, there may be occasions where the extent of the near wake might become an issue; this is an engineering choice that is left to the user.

2.2.3 Code Modifications

The common blocks were changed as follows:
In subroutine FILEJ:

```
/WKC1CM/ WKC1(7),C1(135000),CNW1(29600)
/WKC2CM/ WKC2(7),C2(135000),CNW2(29600)
```

was changed to :

```
/WKC1CM/ WKC1(7),C1(252720),CNW1(972000)
/WKC2CM/ WKC2(7),C2(252720),CNW2(972000)
```

In subroutines VINDCAL1, VINDCAL2, WKC1INT, WKC2INT, and CFD-WAKE:

```
/WKINT/ CINT(3,72000),CNWINT(3,30000)
```

was changed to:

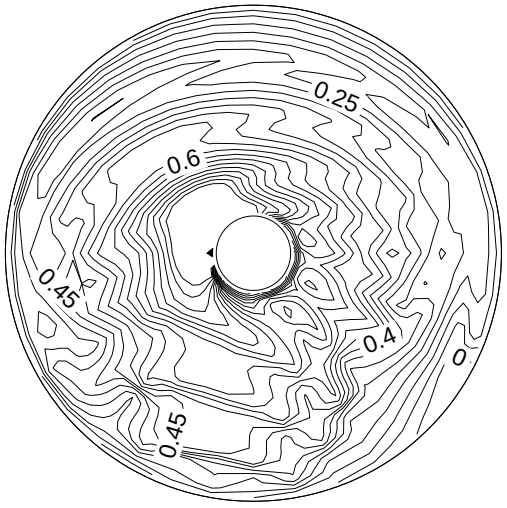
```
/WKINT/ CINT(3,72000),CNWINT(3,110000)
```

In subroutines WAKEC1, WAKEC2, WAKEC1, WAKEN2, WKC1INT, and WKC2INT:

```
/WKC1CM/ ...,CNW(3,97200)
/WKC2CM/ ...,CNW(3,97200)
```

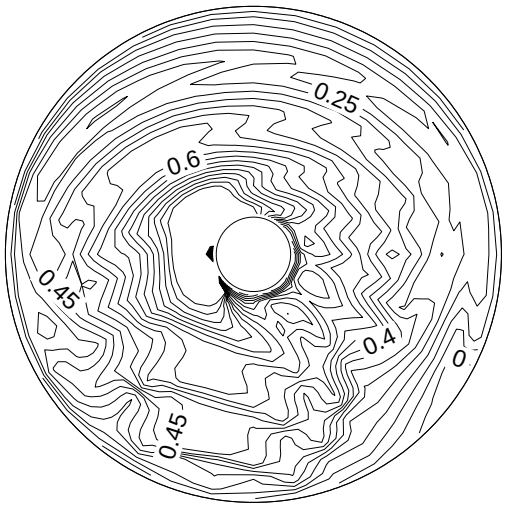
was changed to:

```
/WKC1CM/ ...,CNW(3,324000)
/WKC2CM/ ...,CNW(3,324000)
```



$\psi = 0.0^\circ$

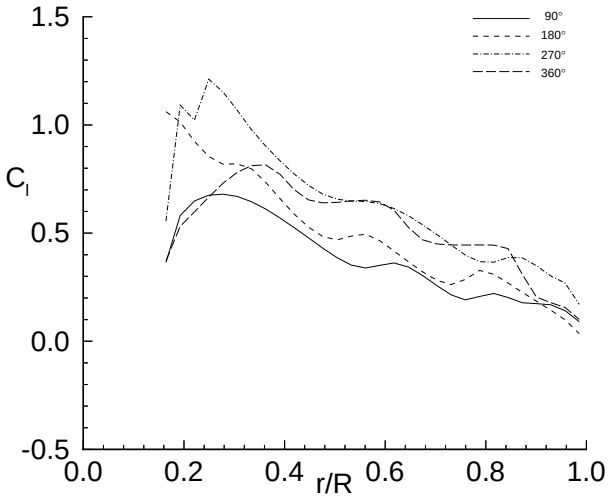
(a) C_l for 10° wake resolution with near wake extent of 20° (KNW = 2)



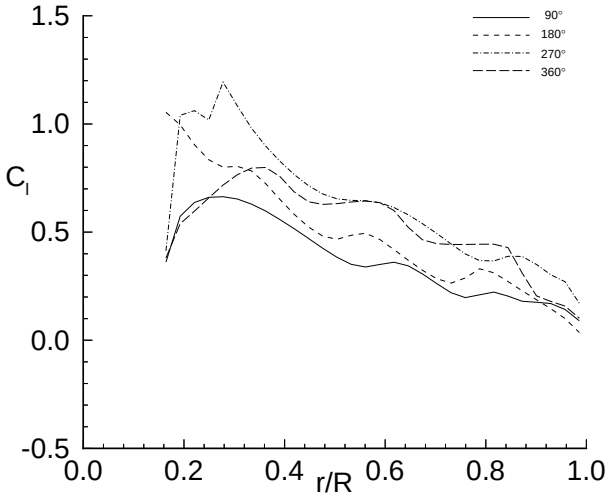
$\psi = 0.0^\circ$

(b) C_l for 10° wake resolution with near wake extent of 90° (KNW = 9)

Figure 2.4: Contours of lift coefficient showing the effect of changing the near wake extent.



(a) C_l for 10° wake resolution with near wake extent of 20° (KNW = 2)



(b) C_l for 10° wake resolution with near wake extent of 90° (KNW = 9)

Figure 2.5: The local lift coefficient as a function of span showing the effect of changing the near wake extent.

In subroutines CHEKR1 and CHEKR2:

```
NWMAX = MRG*MRL*MPSI*MAX0(2,KNW+1)
IF (NWMAX .GT. 97200) GOTO 41
```

was changed to:

```
NWMAX = MRG*MRL*MPSI*MAX0(10,KNW+1)
IF (NWMAX .GT. 324000) GOTO 41
```

With these changes, and using the maximum values of MRA=30, MRG=30, MRL=30, and MPSI=36, the near wake may be extended to 90 ° (*i.e.,* KNW=9 in namelist NLWAKE).

2.2.4 Extensions to High Resolution

The only high resolution variable affected by this change is the array size of the variable CNWINT. This array has been dimensioned to be compatible with the low resolution 90 ° near wake modification. Other common blocks (WKC1CM and WKC2CM) listed in the high resolution subroutines WKC1INT, WKC2INT, VINDCAL1, and VINDCAL2, have been dimensioned so that their size is consistent throughout the code.

2.3 Modifications for Higher Harmonic Control (HHC) of Pitch

2.3.1 Introduction

In the past, there has been considerable interest in the concept of Higher Harmonic Control (HHC) as a means to modify the certain aspects of helicopter behaviors, such as vibratory loads and acoustic signatures (ref [8]). Recently there have been major experimental programs conducted to study the use of HHC to reduce the vibratory loads of forward flight conditions and to reduce the BVI noise levels for descent flight conditions (Ref. [9]). In an attempt to predict or systematically study the effects of HHC, options were added to CAMRAD.Mod1 to include a fixed, user prescribed HHC.

In a typical wind tunnel trim case, the blade pitch in the CAMRAD.Mod1 trim loop is adjusted at the collective (0/rev) and cyclic (1/rev) levels until a trimmed solution is obtained. It is desirable to include in the trim solution, an additional blade pitch that represents HHC. The HHC is

a fixed, open loop quantity added to the existing rigid pitch motion calculated in the "motion iteration" (see Figure 1.2). Therefore, the pitch of the blade will include control inputs (0/rev and 1/rev), blade elastic torsion (due to sources such as pure torsion, pitch-bending coupling, *etc.*), and a prescribed HHC. CAMRAD.Mod1 has two modifications for HHC. These two modifications vary only in their generality and inputs.

2.3.2 HHC Pitch Equations

CAMRAD.Mod1 now calculates the blade pitch input from the following equation:

$$\theta(\psi) = \theta_0 + \theta_{1c}\cos(\psi) + \theta_{1s}\sin(\psi) \; + \theta_{HHC}(\psi) \tag{2.1}$$

where θ_0 is the collective pitch, θ_{1c} is the lateral cyclic pitch, θ_{1s} is the longitudinal cyclic pitch, and θ_{HHC} is the new HHC input.

Two HHC modifications are in CAMRAD.Mod1. The first HHC modification (input via namelist NLHHC), made early in the CAMRAD.Mod1 development process, provides only the capability to model 3/rev, 4/rev, and 5/rev HHC via the following equation:

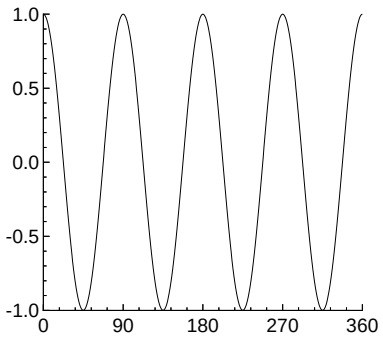
$$\theta_{HHC}(\psi) = term1 + term2 + term3 \tag{2.2}$$

$$term1 = \theta_{coll}\cos(4\psi - 4\phi_{coll}) \tag{2.3}$$

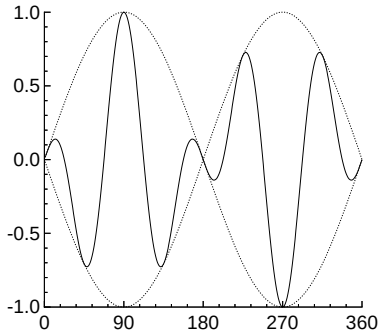
$$term2 = \theta_{lat}\cos(4\psi - 4\phi_{lat})\sin(\psi) \tag{2.4}$$

$$term3 = \theta_{lon}\cos(4\psi - 4\phi_{lon})\cos(\psi) \tag{2.5}$$

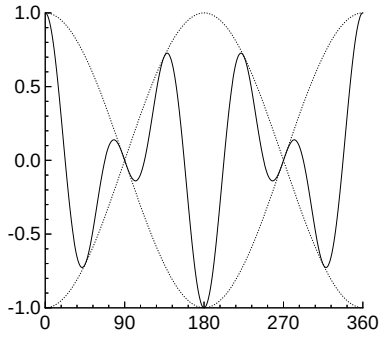
where $\theta_{coll}, \theta_{lat}, \theta_{lon}, \phi_{coll}, \phi_{lat}$, and ϕ_{lon} are input values in degrees. Figure 2.6 illustrates each of the 3 terms in the above equations. In each plot, the values of ϕ_{coll}, ϕ_{lat}, and ϕ_{lon}, have been set to zero to demonstrate each term; these quantities serve only to phase-shift the waveform. The upper plot in Figure 2.6 shows $term1$ in the Equation 2.3. It illustrates a prescribed 1° amplitude, 4/rev HHC pitch input. The lower left plot shows $term2$ in Equation 2.4 (solid line) along with the sine wave "envelope" defined by the term. The lower right plot illustrates $term3$ in Equation 2.5 (solid line) along with the cosine "envelope" defined by the term. Again, both terms are plotted for a 1° amplitude HHC input and the ϕ_{lat} and ϕ_{lon} terms serve to phase shift each waveform. It can be shown that the above equations can be used to generate a pure 3/rev, 4/rev, or 5/rev HHC pitch signal. Equation



(a) term1



(b) term2



c) term3

Figure 2.6: HHC "Terms"

2.2 can also produce "wavelets" that are comprised of 3/rev, 4/rev, and 5/rev HHC components.

In order to have a more flexible HHC input, and to allow the possibility of a form of Individual Blade Control (IBC), the second, more recent modification (input via namelist NLHHC2), uses a truncated Fourier series to represent the HHC pitch. The HHC equation for this modification is as follows:

$$\theta_{HHC}(\psi) = \theta_{HHC,0} + \sum_{n=1}^{12} (A_n \cos(n\psi) + B_n \sin(n\psi)) \tag{2.6}$$

where A_n, B_n, and $\theta_{HHC,0}$ are input values in degrees. The current maximum allowable number of HHC input harmonics is twelve. With Equation 2.6, any HHC waveform may be input approximately by a twelve term Fourier series. It should be noted that zeroth and first harmonics are usually redundant inputs since the rotor in CAMRAD.Mod1, in a typical wind tunnel scenario, is trimmed by adjusting the collective and first harmonics of pitch. If this is the case, the trimmed collective and cyclic pitch values will merely compensate for these input HHC values. They have been included here for completeness and are normally always equal to zero.

Also now included in the CAMRAD.Mod1 code are additional motion terms due to HHC pitch rate and HHC pitch acceleration as calculated by the following equations:

$$\dot{\theta}_{HHC} = \dot{\lambda}(1 + DPS) \tag{2.7}$$

$$\ddot{\theta}_{HHC} = \ddot{\lambda}(1 + DPS)^2 + \dot{\lambda}(DDPS) \tag{2.8}$$

where DPS and $DDPS$ are internal CAMRAD.Mod1 quantities used to account for hub and shaft motion, and $\dot{\lambda}$ is the azimuthal derivative of either Equation 2.2 or 2.6 and $\ddot{\lambda}$ is the second azimuthal derivative of Equation 2.2 or 2.6, depending on the HHC model being used.

2.3.3 Sample Case Discussion

Figures 2.7 (a) and 2.7 (b) show the 10 ° azimuth case without HHC and with the inclusion of a 4/rev, 1 ° amplitude HHC pitch which is a pure cosine wave starting at $\psi = 0$ °. Figures 2.8 (a) and 2.8 (b) show the same information plotted radially at several azimuth locations. In the HHC case, a 4/rev pattern can be seen in the loads due to the 4/rev pitch input (Figure 2.7 (b)). Since the HHC is included in the blade motion, the 4/rev loading

is not necessarily in phase with the HHC pitch input. This can be seen in Figure 2.7 (b); the 4/rev loading is not a pure cosine wave starting at $\psi = 0^\circ$ as is the HHC input. This case demonstrates that HHC can greatly impact the loads for a given rotor.

2.3.4 Code Modifications

In subroutines INPTR1 and INPTR2, changes were made to read in both HHC option parameters via namelist NLHHC for the first modification and NLHHC2 for the second modification. The parameters in both namelists are initialized to zero and are converted internally to radians after being read. The input values are thus in degrees. The first set of modification parameters are saved in the common block HHC1 for rotor-1 and in common block HHC3 for rotor-2. The second set of modification HHC parameters are saved in common block HHC2 for rotor-1 and in common block HHC4 for rotor-2. The HHC equations are programmed in the subroutines HHCTRM1 and HHCTRM2 for rotor-1 and rotor-2, respectively. The pitch, pitch rate, and pitch acceleration terms due to HHC are included in the blade motion subroutines MOTNB1 and MOTNB2 for rotor-1 and rotor-2, respectively.

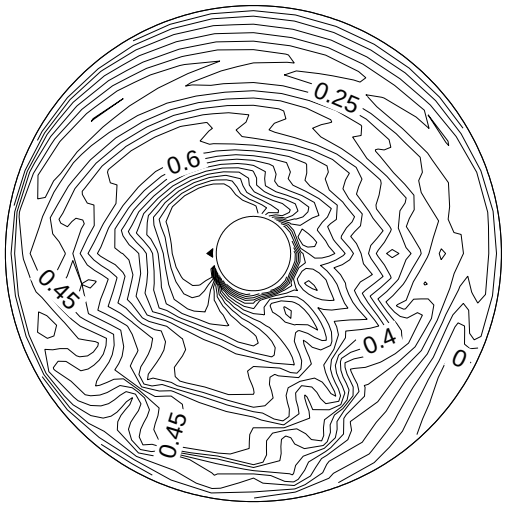
2.3.5 Extensions to High Resolution

Since the blade motion in HIRES is obtained from the low resolution blade motion determined in CAMRAD.Mod1, the HHC modifications discussed here are automatically included in the HIRES reconstruction.

2.4 Modifications for Aerodynamic Sweep Effects

2.4.1 Introduction

It is widely known that, in fixed-wing aircraft, swept wings have advantages in reducing the compressibility effects of high speed flight (ref [10]). In the case of rotorcraft, the blade tips are traveling at high subsonic Mach numbers, and thus encounter compressibility effects. Many rotorcraft manufacturers are or have been using some form of swept tip rotor blade design. The original version of CAMRAD did not apply any models to account for planform sweep. To study aerodynamic effects of mildly swept planforms, a modification was made to CAMRAD.Mod1 to model these effects. This modification provides a means by which airfoil characteristics (*i.e.,* lift,



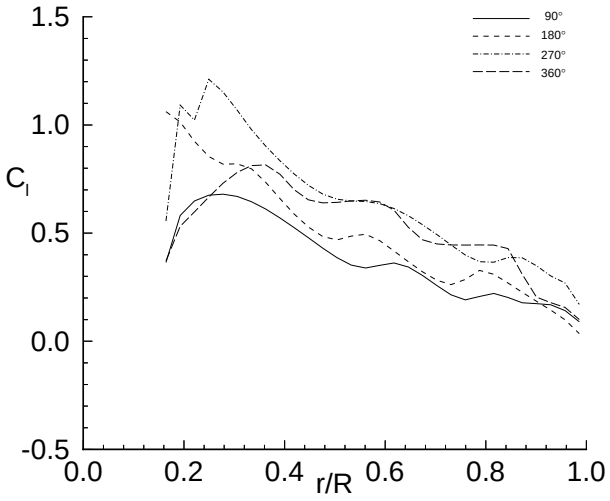
$\psi = 0.0^\circ$

(a) C_l for 10° wake resolution without HHC input.

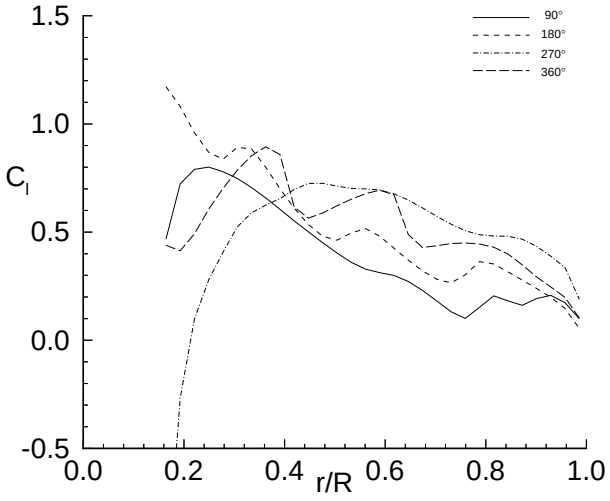
$\psi = 0.0^\circ$

(b) C_l for 10° wake resolution with a 4/rev, 1° amplitude HHC pitch input.

Figure 2.7: Contours of local lift coefficient C_l over the rotor disk showing effect of HHC inputs.



(a) C_l for 10 ° wake resolution without HHC input.



(b) C_l for 10 ° wake resolution with a 4/rev, 1 ° amplitude HHC pitch input.

Figure 2.8: The local lift coefficient as a function of span showing the effect of HHC inputs.

drag, and moment coefficients) calculated in the "circulation iteration" (see Figure 1.2) for swept, yawed flow conditions can be analytically related to unswept, unyawed conditions. Since tabulated yawed, swept airfoil characteristics are not normally available, this analytical relation facilitates use of available 2-D tabulated airfoil characteristics. Note that these modifications are only included in the aerodynamics of the rotor, and therefore are included in the aerodynamic forcing functions for the rotor; no modifications have been made to alter the blade dynamics to account for effects of swept planforms (that is, assumptions such as a straight elastic axis, *etc.* are still in place).

2.4.2 Angle of Attack Correction

Since airfoil data is tabulated for 2-D unswept sections, it is convenient to relate section properties for a yawed, swept planform to those of an unyawed, unswept section so that these tabulated tables may be used for yawed, swept blade sections. CAMRAD and CAMRAD.Mod1 already account for yawed, unswept flow effects as follows:

$$C_l(\alpha) = \frac{C_{l,2d}(\alpha_{t1}, M_t)}{\cos^2(\Lambda)} \tag{2.9}$$

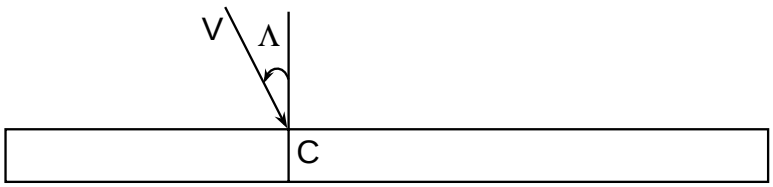
$$C_d(\alpha) = \frac{C_{d,2d}(\alpha_{t2}, M_t)}{\cos(\Lambda)} \tag{2.10}$$

$$C_m(\alpha) = C_{m,2d}(\alpha_{t1}, M_t) \tag{2.11}$$

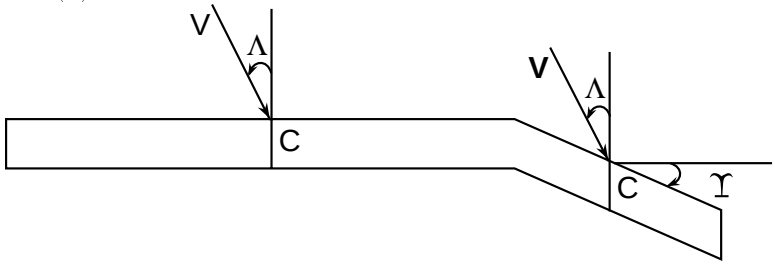
$$\alpha_{t1} = \alpha \cos(\Lambda) \cos(\Lambda) \tag{2.12}$$

$$\alpha_{t2} = \alpha \cos(\Lambda) \tag{2.13}$$

where Λ is the yaw angle between the flow and a section perpendicular to the spanwise reference line of the blade, α_{t1} and α_{t2} are angles of attack to be used in the airfoil table interpolation, M_t is the Mach number to be used in the airfoil table interpolation, α is the calculated angle of attack for the 2-D section, and quantities with the subscript $2d$ are values found by the airfoil table interpolation. The airfoil tables are interpolated to determine the $C_{l,2d}, C_{d,2d}$, and $C_{m,2d}$ at the angle of attack, α_{t1} (or α_{t2} for the drag) and at the Mach number, M_t. Once these 2-D values are known, the above equations relate the unyawed, unswept values to the desired yawed, unswept values to be used in the analysis. Note that the cosines in the denominators of the lift equation arise from dynamic pressure differences between the



(a) Non-swept planform with flow yawed at angle Λ.



(b) Swept planform of sweep angle ϒ and flow yaw angle Λ.

Figure 2.9: Non-swept and swept blade planforms.

yawed and the unyawed flows. The single denominator of the C_d equation results from the assumption that the total drag is in the yawed flow direction (a cosine factor in the numerator has canceled out one of the cosine factors in the denominator). The cosine factors in the angles of attack, α_{t1} and α_{t2}, used in the airfoil table interpolation come from two sources. One cosine comes from the angle of attack difference between the yawed, unswept and unyawed, unswept flows. This is the origin of the cosine in the angle of attack in the α_{t2} equation above. The second cosine in the α_{t1} equation above arises from the "swept wing equivalence assumption" (Ref. [11]). This cosine accounts for the difference in lift curve slopes between yawed, unswept and unyawed, unswept flows. In the context of Ref. [1] and [11], a "swept wing" refers to the entire, straight blade, being angled (skewed) to the freestream (Figure 2.9 (a)).

To model swept planform effects on the aerodynamics, the "swept wing equivalence assumption" is modified to include the local sweep angle effect on the angle of attack and Mach number used in the table interpolation. This effect does not affect the dynamic pressure portion of the equations 2.9 - 2.13 above (*i.e.,* the denominators remain unchanged) since the modification merely relates the yawed, swept properties to the unyawed, unswept properties.

The angle of attack modification is made solely to relate the yawed,

swept flow to an equivalent unyawed, unswept flow. This technique is used so that the airfoil tables for a 2-D airfoil section (perpendicular to the reference span) may be retained and such that the 2-D airfoil tables remain independent of the yaw and/or sweep angles. Without this modification, the airfoil tables would necessarily be a function of not only angle of attack and Mach number, but also yaw angle and sweep angle. Figure 2.9 (b) illustrates the sweep of a planform and the meaning if yaw angle versus sweep angle. The sweep angle actually required for the analysis is the sweep of the quarter-chord line if the section is tapered.

To include modification for the angle of attack, the CAMRAD.Mod1 Equations 2.9 - 2.13 shown above are changed to the following:

$$C_l(\alpha) = \frac{C_{l,2d}(\alpha_{t1}, M_t)}{\cos^2(\Lambda)} \tag{2.14}$$

$$C_d(\alpha) = \frac{C_{d,2d}(\alpha_{t2}, M_t)}{\cos(\Lambda)} \tag{2.15}$$

$$C_m(\alpha) = C_{m,2d}(\alpha_{t1}, M_t) \tag{2.16}$$

$$\alpha_{t1} = \alpha \cos(\Lambda + \Upsilon) \cos(\Lambda) \tag{2.17}$$

$$\alpha_{t2} = \alpha \cos(\Lambda) \tag{2.18}$$

where Υ is the sweep angle of the quarter chord line of the blade with respect to the reference span line. The reference span line has the same definition as in the original CAMRAD version. Note that due to the "swept wing equivalence assumption", only one of the cosine terms in the equations above is affected. Next, the calculation of the Mach number to use in the table interpolation is discussed.

2.4.3 Mach Number Correction

As discussed previously, the airfoil tables are interpolated to find a value of $C_{l,2d}, C_{d,2d}$, and $C_{m,2d}$ at a particular angle of attack and Mach number. Then, the equations above are applied to calculate the C_l, C_d, and C_m used in the CAMRAD.Mod1 analysis. In the previous subsection, the modification to the angle of attack used in the table interpolation was discussed. This subsection discusses the Mach number modification.

It is well known from swept wing analysis that there is a compressibility relief due to local sweep of a planform. The Mach number modification is therefore cast in the form of a compressibility relief term referenced to

the original velocity vector at the section. Swept wing theory, along with a high aspect ratio assumption, implies that the correct Mach number to use in aerodynamic calculations is the Mach number perpendicular to the quarter chord line. Thus the Mach number to be used in the airfoil table interpolation of 2-D loading must be modified to account for such a relief. For the case of no sweep, the Mach number calculation, is as follows (this is the form used by the original version of CAMRAD):

$$M_t = M_\perp = \left[\frac{\sqrt{(U_t^2 + U_p^2)}}{a_\infty} \right] M_{corr} \tag{2.19}$$

where M_t is the Mach number to be used in the table interpolation, $M_\perp$ is the Mach number in a plane perpendicular to the straight blade (*i.e.,* the straight elastic axis), U_p is the velocity perpendicular to the hub plane at the current section, U_t is the velocity parallel to the hub plane at the current section, a_∞ is the speed of sound as calculated internally in CAMRAD.Mod1, and M_{corr} is a user input constant (input as a function of blade span) that could be used to account for any desired constant compressibility relief. For a straight, unswept planform, this equation is consistent with the swept wing analysis. In order to account for a swept planform, however, this equation must be modified. The modification involves calculating the total Mach number from the value of $M_\perp$ as follows:

$$M_{total} = \frac{M_\perp}{\cos \Lambda} \tag{2.20}$$

then, calculating the Mach number normal to the quarter chord (see Ref. [10]), as follows:

$$M_n = M_{total} \cos(\Lambda + \Upsilon) \tag{2.21}$$

Combining these two equations, the Mach number to be used in the 2-D airfoil table lookup of properties is:

$$M_t = \frac{M_\perp \cos(\Lambda + \Upsilon)}{\cos(\Lambda)} \tag{2.22}$$

2.4.4 Sample Case Discussion

Figure 2.10 shows the lift coefficient contours for 2 cases. Figure 2.10 (a) is the 10 $^\circ$ azimuth step case with no sweep and Figure 2.10 (b) is the same

case except that 30 ° of aft sweep outboard of r/R = 0.81 is included. Figure 2.11 illustrates the same information, plotted radially at several azimuth locations. For this particular case, the sweep correction model has a very small effect.

2.4.5 Code Modifications

In subroutines INPTR1 and INPTR2, the following were added:

```
REAL SWPLO(30),SWPHI(100)
NAMELIST /NLSWP/ SWPLO
COMMON /SWPCM1/ SWPLO,SWPHI  (added to INPTR1)
COMMON /SWPCM2/ SWPLO,SWPHI  (added to INPTR2)
```

A read of the namelist NLSWP was added to INPTR1 and INPTR2 after the read of namelist NLBED (to be discussed in a later section). Input variables are converted to radians after input and are saved in the common blocks SWPCM1 for rotor-1 and in SWPCM2 for rotor-2. All SWPLO and SWPHI quantities are initialized to zero before reading namelist NLSWP. Once the low resolution sweep quantities are input, the subroutines INITHR1 and INITHR2 interpolate these quantities to the required high resolution quantities for use in HIRES. In subroutines AEROS1 and AEROS2, changes are as follows:

add the lines:

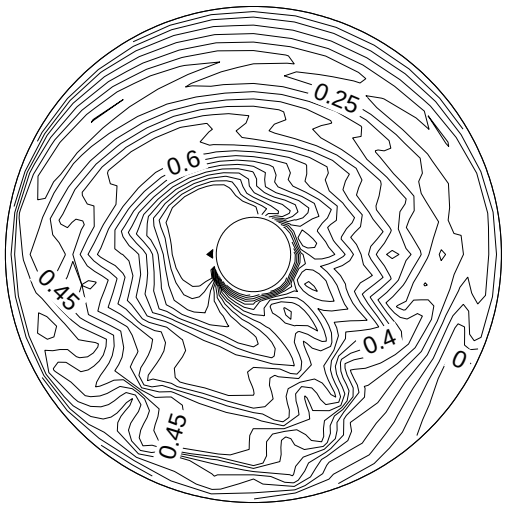
```
REAL SWPLO(30),SWPHI(100)
COMMON /SWPCM1/ SWPLO,SWPHI  (added to AEROS1)
COMMON /SWPCM2/ SWPLO,SWPHI  (added to AEROS2)
```

change the lines:

```
AEL = ADL*COSLSQ
AED = ADD*COSL
AEM = ADM*COSLSQ
```

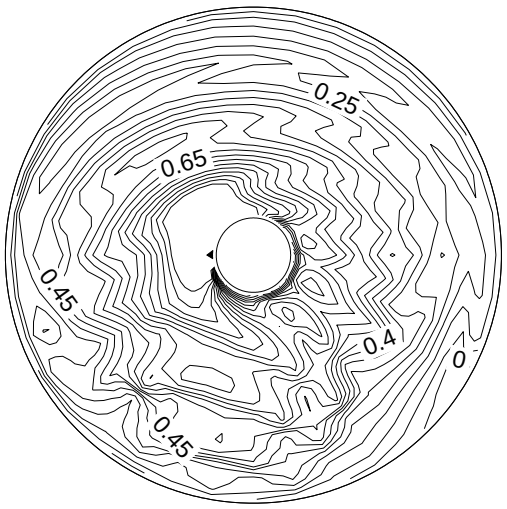
to the following:

```
YAWANG = ACOS(COSL)
COSL2  = COS(YAWANG + SWPLO(IR))
COSL3  = COSL*COSL2
```



$\psi = 0.0\ ^\circ$

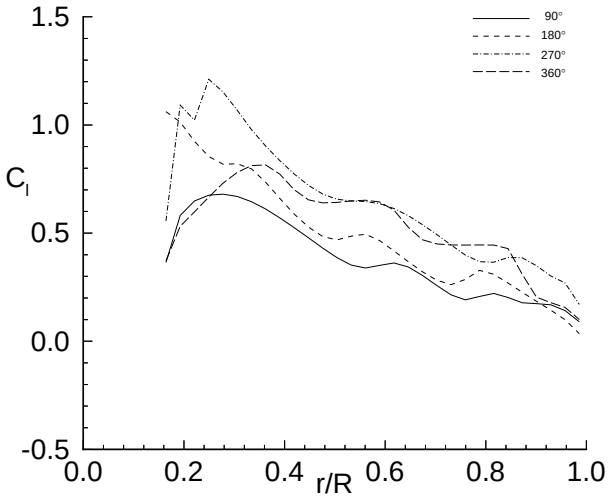
(a) C_l for $10\ ^\circ$ wake resolution for an unswept planform.



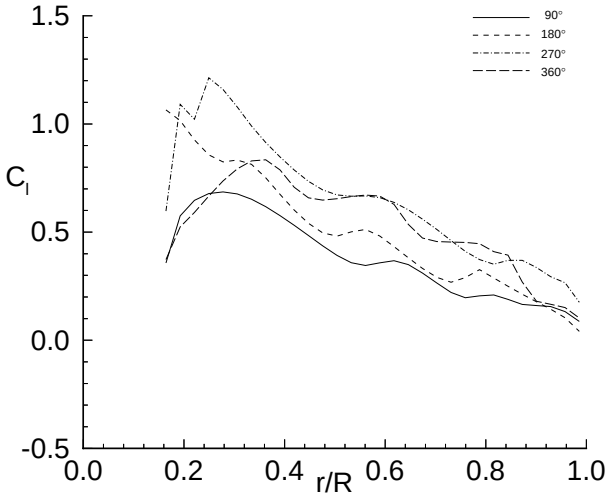
$\psi = 0.0\ ^\circ$

(b) C_l for $10\ ^\circ$ wake resolution with $30\ ^\circ$ of sweep outboard of $\frac{r}{R} = 0.81$

Figure 2.10: Contours of local lift coefficient C_l over the rotor disk showing effect of aerodynamic sweep correction.



(a) C_l for 10 ° wake resolution for an unswept planform.



(b) C_l for 10 ° wake resolution with 30 ° of sweep outboard of $\frac{r}{R} = 0.81$

Figure 2.11: The local lift coefficient as a function of span showing the effect.

```
IF (COSL .NE. 0.) THEN
   ML = ML*COSL2/COSL
   MD = MD*COSL2/COSL
   MM = MM*COSL2/COSL
ENDIF
AEL = ADL*COSL3
AED = ADD*COSL
AEM = ADM*COSL3
```

2.4.6 Extensions to High Resolution

The low resolution sweep modifications discussed in this section are also applied in HIRES. No additional user input is required for this to occur. The low resolution inputs from the variable SWPLO are internally interpolated to the high resolution radial stations input by the user in the array RAEINT of namelist NLHIRES.

2.5 Modification to Airfoil Tables

2.5.1 Introduction

The number of and size of the airfoil tables input to CAMRAD.Mod1 is limited. In order to input more C81 airfoil tables and/or more angles/Mach numbers per table, several common blocks and several IF statements were changed. In the airfoil table preparation program AIRFOIL (see Figure 1.2), the C81 airfoil tables are read and converted to a "CAMRAD airfoil table" ("airfoil.tab" in Figure 1.2) format. This format allows for efficient interpolation of the airfoil aerodynamic information during CAMRAD.Mod1 execution. For some rotorcraft, multiple tables need to be read and used. This modification effectively increases the number of and size of the input airfoil tables.

2.5.2 Code Changes

In AIRFOIL and in the subroutine AEROT of AIRFOIL, the common block TABLES was changed such that the dimensions of the variables CLT, CDT, and CMT were increased from 5000 to 10000. Also, the IF statement:

```
IF (NA(NAB)*NM(NMB)*NRB .GT. 5000) ICHECK = 1
```

was changed to:

```
IF (NA(NAB)*NM(NMB)*NRB .GT. 10000) ICHECK = 1
```

in the airfoil preparation program.

In CAMRAD.Mod1, similar changes were made. Common blocks A1TABL and A2TABL in the subroutines AEROT1, AEROT2, AET1INT, AET2INT, FILER, INPTA1, and INPTA2, were changed such that the dimensions of the variables CLT, CDT, and CMT were increased from 5000 to 10000. Also, the IF statement:

```
IF (NMAX .GT. 5000) GOTO 12
```

was changed to:

```
IF (NMAX .GT. 10000) GOTO 12
```

in subroutines INPTA1 and INPTA2.

2.5.3 Extensions to High Resolution

Since the common blocks in subroutines AET1INT and AET2INT were changed, no other user intervention is needed for application to the high resolution part of the code (HIRES).

2.6 Modification to Motion Convergence

2.6.1 Introduction

When CAMRAD.Mod1 fails to converge to a trimmed condition, many times, an inner loop is the cause of convergence failure. For example, if the circulation loop (see Figure 1.2) diverges, most likely the trim loop will also diverge. To assist in circulation loop convergence, a lag (relaxation) factor is employed in CAMRAD.Mod1. However, in some instances, the trim divergence is caused by motion loop divergence. In the original version of CAMRAD, there is no relaxation factor in the motion loop to assist convergence. To help motion convergence in these situations, a relaxation factor was added inside the motion loop. This relaxation factor was added to the rotor forcing function in order to make the trim convergence more robust. The relaxation factor is a user specified factor input to linearly lag

the rotor forcing function between successive motion iterations. The form of the relaxation is as follows:

$$F(k,\psi) = F_{old}(k,\psi) * (1 - \text{FACTM}) + F(k,\psi) * (\text{FACTM}) \qquad (2.23)$$

where F is the forcing function, F_{old} is the forcing function from the previous revolution, FACTM is the user specified motion relaxation factor, k is the mode shape index, and ψ is the current azimuth location. This equation is utilized in subroutines INRTM1 and INRTM2 for rotor-1 and rotor-2, respectively. The same relaxation factor is used for both rotors. For FACTM = 1.0, Equation 2.23 produces the same result as the original CAMRAD motion iteration.

2.6.2 Code Modifications

The relaxation factor has been added to the namelist NLTRIM, which is read by the subroutine, INPTN. The default is FACTM = 1.0, which produces no relaxation in the forcing function, as was implemented in the original version of CAMRAD. A common block, FORCCM, was added to subroutines INITR1, INITR2, INPTN, INRTM1, INRTM2, and PRNT. The common block variables are OLDF1(16,36), OLDF2(16,36), and FACTM. Coding was added to subroutines INITR1 and INITR2 to initialize the vectors OLDF1 and OLDF2 to zero. Coding was added to subroutine INPTN to include FACTM in the namelist NLTRIM. Coding was added to the subroutine PRNT to include a listing of the value of FACTM in the "INPUT DATA" section of the printed output, if the section is requested.

2.6.3 Extensions to High Resolution

Since HIRES does not re-trim the rotor, or update blade motion in any way, this modification has no effect in HIRES.

2.7 ROTONET/WOPWOP Interface

2.7.1 Introduction

System noise predictions are frequently used to determine the effects of design changes in parametric studies. Since many configurations are evaluated, short computer run times are essential. However, accuracy is also

needed, which necessitates high-quality airloads. As a part of the NASA Langley rotor noise prediction efforts, a method was developed for connecting the airloads calculations of CAMRAD.Mod1 to the ROTONET rotorcraft systems noise code (Ref. [6]).

This method includes several new subroutines and input parameters to CAMRAD.Mod1. Execution of CAMRAD.Mod1 may yield two ASCII text files for use in ROTONET. The first file, named ROTPARAM.DAT, contains standard ANOPP control statements for defining various PARAMETER inputs to the ROTONET functional modules. The second file, named ROTABLES.DAT, has table members which provide rotor aerodynamic and dynamic information (normally computed by ROTONET modules LRP, RWG, RIN, RRD, and RLD) in the correct form for use by the ROTONET source noise modules LRN, RTN, and RBN. The first file is intended to be "cut-and-pasted" into a ROTONET input deck which executes LRN, RTN, and/or RBN. The second file is a self-contained ROTONET input job, and when input to ROTONET, will UNLOAD the table members into the CAMROT.WRK library file. This library file is then LOADed into the ROTONET input file which executes source noise modules, thereby the source noise modules in ROTONET can then utilize the airloads calculated by CAMRAD.Mod1.

The CAMRAD.Mod1 notation for two-rotor vehicles is used; that is, "rotor-1" and "rotor-2". For conventional helicopters, the main rotor is rotor-1 and the tail rotor is rotor-2. For tandem helicopters, the forward rotor is rotor-1 and the rear rotor is rotor-2. For side-by-side rotors, such as tiltrotors, rotor-1 is the starboard rotor and rotor-2 is port rotor. Variables with "R1" and "R2" in the names are for rotor-1 and rotor-2, respectively.

In addition to the ROTONET information, 4 files, two for each rotor, are output for use in the rotor tone noise code WOPWOP. The first file for each rotor, named WOPWOP-R1.DAT and WOPWOP-R2.DAT (rotor-1 and rotor-2, respectively) contain the WOPWOP input namelist, INPUT. The second file for each rotor, named WOPFORCE-R1.DAT and WOPFORCE-R2.DAT (rotor-1 and rotor-2, respectively) contain the vertical and inplane sectional forces at the CAMRAD.Mod1 radial and azimuthal locations.

2.7.2 Code Modifications

A variable, NOISFL, was added to the NLCASE namelist in CAMRAD.Mod1 as a switch to turn on/off the output of the ROTONET/WOPWOP information. If NOISFL = 0, then no information

for these programs is output. If NOISFL = 1, then information is output. NOISFL was also added to the common block CASECM in the main program, CAMRAD, and in the following subroutines: FILEE, FILER, FILEV, FLUT, INPTN, INPTO, PRNTC, PRNTJ, ROTNET, STAB, STABD, STABE, TRAN, and TRIM. If NOISFL = 1, the subroutine TRIM calls the new subroutine ROTNET to calculate and output the ROTONET/WOPWOP information. In addition to ROTNET, three new subroutines were added for use by ROTNET: RMTN1, RMTN2, and HAVAR. To draw on information calculated already in the main part of the code, a new common block, RTNCM, was added to these subroutines: PERFR1, PERFR2, PRNTC, and ROTNET.

2.7.3 Known Caveats

The use of these modifications is strongly NOT recommended as they do not include higher harmonic loads and may produce misleading results. These modifications as such have not been exercised nor have they been fully tested.

2.7.4 Extensions to High Resolution

Since these modifications are intended to output low resolution information for use in other codes, these modifications have no bearing on the HIRES portion of the code.

2.8 CFD (FPRBVI) Interface

2.8.1 Introduction

Use of most CFD codes for rotor problems requires *a priori* knowledge of aircraft trim, rotor dynamics, and wake aerodynamics. The standard method for obtaining these quantities is to use CAMRAD.Mod1 to perform the usual trim and performance calculations, and then output quantities ready for use in isolated-blade CFD codes such as FPRBVI. A common method for transferring the aerodynamic environment calculations to CFD codes is through a "partial" angle of attack table. This partial angle sums the effects of all blade motions and fluid velocities, in a lifting-line form, less the effect of the reference blade's own near wake computed explicitly in the CFD analysis within the computational domain (hereafter denoted by the

term "CFD box"; see Figure 2.12). The CFD code reads this table, then uses it to modify the velocity field through which the blade travels. If modeling BVI events, detailed information about the vortex wake is required by the CFD (FPRBVI) analysis. A method to calculate the required vortex wake information is implemented using a non-rotating "BVI box" (see Figure 2.13) that surrounds the rotor (discussed later). Another method for modeling the aerodynamic environment in the CFD analysis is to account for blade dynamic motions (including pitch inputs) and wake-induced velocities separately. The careful use of these effects allows for a more accurate calculation, including such effects as pitch rate. Upon computing the airloads for one revolution or more, the CFD code may output lift and moment coefficients. The file containing these coefficients can then be used by CAMRAD.Mod1 to modify the airloads distribution (lift coefficient only) used in computing the aircraft trim, wake response, and rotor dynamic response.

This work is based in part on work performed by industry on contract to NASA Langley. Numerous updates, corrections, and features have been added by NASA Langley to improve the quality and quantity of information provided to FPRBVI (see Ref. [4]). Several of these features include (1) a new blade-wake coordinate transform for rigid blade cases, (2) a new elastic blade motion interface to output elastic motion information for the FPRBVI analysis, (3) a new direct blade motion modeling interface to FPRBVI, (4) a new interface to include a vortex rollup model, a vortex multi-core model and (5) a new interface to pass tip and secondary vortex trajectories, multi-core core properties, vortex strengths, and vortex locations relative to the CFD and BVI boxes.

2.8.2 Code Modifications

The main routine, CAMRAD, calls the subroutine INPTN which reads the namelist NLTRIM. In the NLTRIM namelist, the variable OPREAD(2) is used as a switch to enable reading of the CFD input namelist, NLCFD, after the read of namelist NLROLL. At present, the CFD interface is only applicable to rotor-1. If OPREAD(2) = 2, CAMRAD.Mod1 expects to read NLCFD after NLROLL. Also, if OPREAD(2) = 2, the subroutine INPTCFD reads the namelist NLCFD which contains the variables OPCFD, OPBVI, PHICFD, RDB(6), BDB(6), and OPMOTN. These variables are listed and described in Chapter 5.

Once the variables are read via the NLCFD namelist, they are stored in the common block CFDDATA. This common block has been added

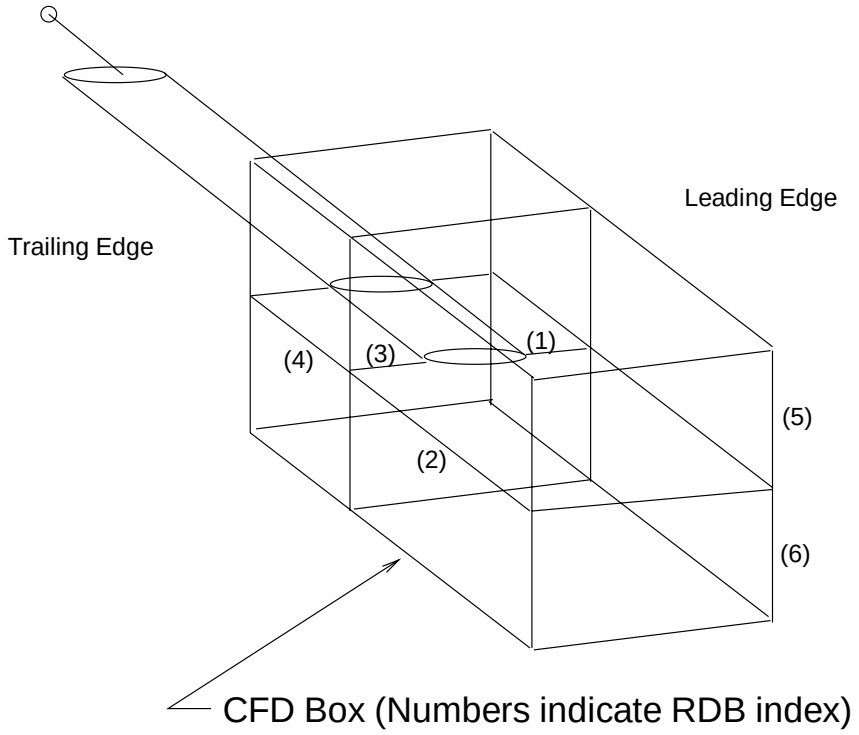


Figure 2.12: CFD Box definitions used in CAMRAD.Mod1 and FPRBVI.

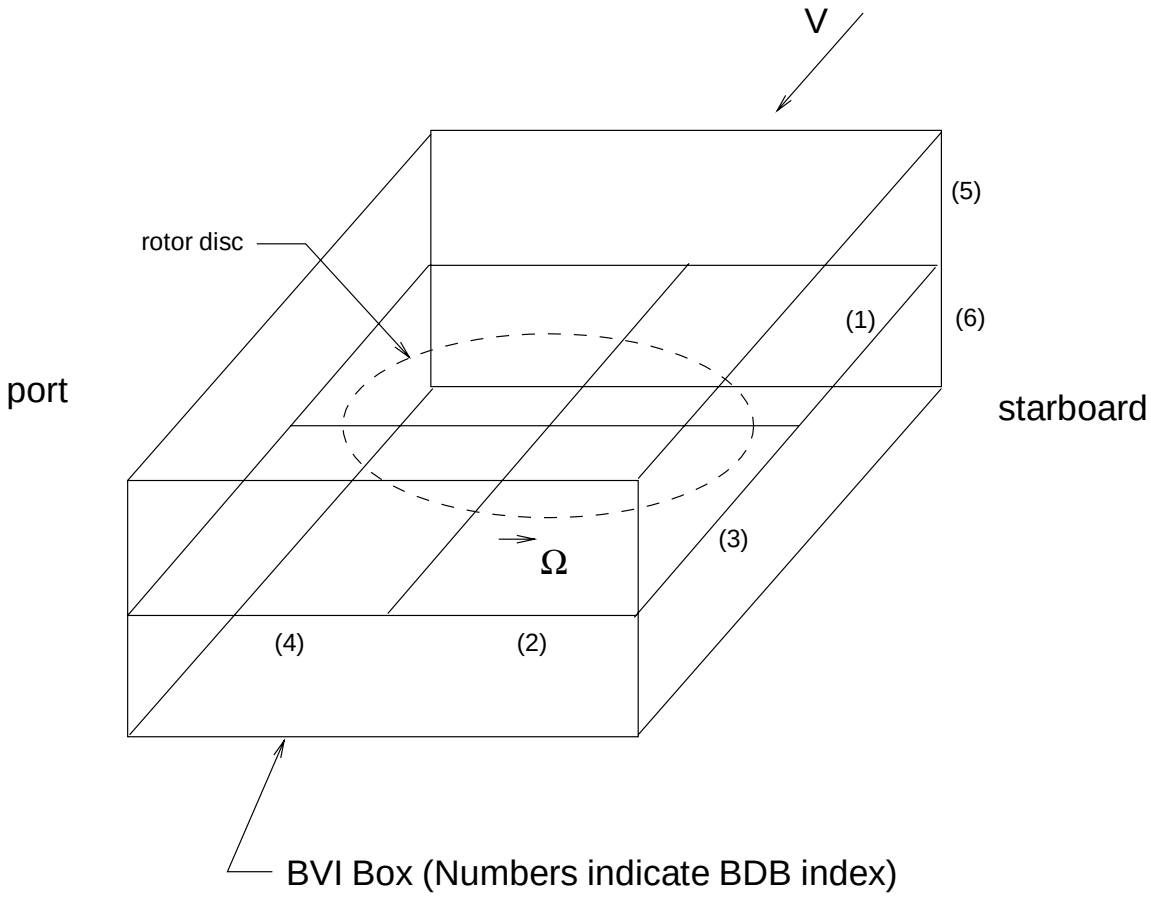


Figure 2.13: BVI Box definitions used in CAMRAD.Mod1 and FPRBVI.

to the executive program, CAMRAD, and to the subroutines AEROF1, AERBED1, BVIBOX, CFDAERO, CFDBOX, CFDWAKE, GEOMBVI, INPTCFD, ROTNET, TRIM, and WAKEC1.

If the variable OPCFD = 1, subroutine TRIM calls the subroutine WAKEC1 at the end of the trim loop, such that it will recalculate the influence coefficients excluding all wake elements not in the CFD box. In order to test if a particular wake segment is in or out of the CFD box, WAKEC1 has been modified such that calls to the subroutines VTXL, VTXL2, and VTXS include arguments to enable or disable testing based on the type of segment. For example, the CFD box only extends around the reference blade; therefore, elements from other blades need not be tested for being in or out of the CFD box. (That is, the CFD box includes only wake elements in the near wake of the reference blade since the near wake is included implicitly in the CFD code.) Both subroutines, VTXL and VTXL2, call the subroutine CFDBOX, which does the actual vortex segment testing for the inclusion in the CFD box. Upon return from WAKEC1, subroutine TRIM calls the subroutine, CFDWAKE, which calculates a velocity parallel and perpendicular to the hub plane for each blade section using the recalculated influence coefficients. This newly calculated velocity is subtracted from the previously stored "full" velocities parallel and perpendicular to the hub plane at the blade section to yield the "partial velocities". These new partial velocities are used to calculate the required "partial angles of attack", α_p, as a function of radius and azimuth. These $\alpha_p(r, \psi)$ are written to a file called ALPHAP.DAT to be read by an external CFD code such as FPRBVI.

Including only the partial angles in the CFD code is appropriate when there are not BVI events. This is because in "non-BVI" flight conditions, the vortices are sufficiently far from the blades that usage of only the partial angles is sufficient. However, in flight conditions were there are significant BVI events, it is necessary to model the blade vortex passages in a more accurate manner. For this purpose, the full potential rotor code, FPRBVI, has a method for directly computing the downwash at the blade due to tip vortex segments generated by an external free wake model. CAMRAD.Mod1 has an option to add the tip vortex segment information to the ALPHAP.DAT file in the form of a wake table containing vortex segment endpoints and strength. In addition to the option OPCFD = 1, if the variable OPBVI = 1, an additional test is done on each tip vortex segment (via a call to the subroutine BVIBOX from the subroutine CFDBOX), in order to determine whether or not that vortex segment is in the BVI box. If the segment is inside the BVI box, the velocity contribution of the segment is also re-

moved from the "partial" inflow as is done if a segment were in the CFD box. The velocity contribution is removed to avoid "double-counting" of the vortex influence since the velocity due to the vortex will be included in the CFD code. This tip vortex wake table includes all vortex end point locations, strengths of each endpoint, and a flag used by FPRBVI to determine which elements to use in its own BVI calculations. In addition, the tip vortex wake information (including the secondary vortex) is written to a file called "ALLWAKE.DAT", which tabulates the tip vortex trajectory, core properties, strength and location relative to the CFD and BVI boxes.

If OPMOTN = 0, the tip vortex wake table is written out with segment endpoint position defined relative to the flapped blade position to account for the effect of flap displacement on blade-vortex miss distance. This is accomplished by subroutine GEOMBVI. If OPMOTN = 1, the tip vortex wake table is written out with segment endpoint position defined relative to the unflapped blade position. The effect of flap displacement on blade-vortex miss distance must then be modeled directly in the FPRBVI calculations using the information in the MOTION.DAT file.

To allow direct modeling of blade motion, both rigid and elastic, in subsequent CFD calculations, rigid blade motion harmonics and elastic corrections are written to the MOTION.DAT file. This file contains the rigid flap, rigid lag, and rigid pitch motion harmonic coefficients as well as the additional blade elastic flap and pitch deflections needed at each azimuth and radial station to reconstruct the blade position. (At present, the elastic lag is not included in the output of this file.) The total blade flap deflection β_{total} at a particular azimuth and ψ can be reconstructed from the rigid flap deflection, $\beta_r(\psi)$, plus an elastic correction $\delta\beta(r,\psi)$,

$$\beta_{total}(r,\psi) = \beta_r(\psi) + \delta\beta(r,\psi) \tag{2.24}$$

where

$$\beta_r(\psi) = \beta_0 + \beta_{1c}\cos\psi + \beta_{1s}\sin\psi \tag{2.25}$$

If OPMOTN = 1, the partial angle-of-attack table written to the ALPHAP.DAT file is replaced with a table of the wake-induced partial inflow at the blade, as a function of radius and azimuth. FPRBVI can then use this wake induced velocity, the shaft angle, the blade rigid pitch inputs, and the rigid flap motion to reconstruct the aerodynamic environment experienced by the reference blade. This method of modeling the flow with

FPRBVI allows modeling of phenomena such as pitch rate effects that are not contained in the original method using α_p information.

In addition to the output file ALPHAP.DAT, the subroutine CFDWAKE creates a file named CAMAERO.DAT. This file is one of two files needed if the option to rerun CAMRAD.Mod1 using the externally generated lift coefficients. The other file needed is CFDAERO.DAT which is generated by the CFD analysis. The option to rerun CAMRAD.Mod1 using a combination of externally generated lift coefficients and internally calculated values for parts of the blade not included in the external calculation, is begun by choosing the option OPCFD = 2 for the rerun. This variable choice forces CAMRAD.Mod1 to read the files CAMAERO.DAT and CFDAERO.DAT via a call to the subroutine, CFDAERO. Both files contain lift coefficients as a function of radial and azimuthal location. These data are stored in the arrays CLOLD and CLEXT, respectively, which are in turn stored in the common block EXTAERO. The common block EXTAERO has been added to the subroutines AEROF1, AERBED1, and CFDAERO and contains the arrays CLOLD(30,36) and CLEXT(30,36). If OPCFD = 2, during the trim process, the lift coefficient is replaced by the formula:

$$Cl(r,\psi) = C_l(r,\psi) - C_{l,old}(r,\psi) + C_{l,external}(r,\psi) \tag{2.26}$$

where $Cl(r,\psi)$ is the currently calculated lift coefficient, $C_{l,old}(r,\psi)$ is the lift coefficient from the previous CAMRAD.Mod1 execution, and $C_{l,external}(r,\psi)$ is the lift coefficient from the external CFD analysis. At the end of the run, the CFD code is rerun, if desired, and the loop repeated in a open loop manner as the user deems necessary.

2.8.3 Known Caveats

It should be noted that when using the Trailed Wake Algorithm (TWA) in the Indicial Aerodynamics, partial angles computed will be erroneous since the TWA, as implemented, can not truncate the near wake to exclude vortex elements inside the CFD box. Thus, it is not recommend at this time to execute the indicial aerodynamics option and the CFD interface option together.

2.8.4 Extensions to High Resolution

These modifications are intended to output low resolution information for use in the external CFD codes. Therefore, they have no bearing on the

HIRES portion of the code.

2.9 Modification for Tunnel/Fuselage Corrections

2.9.1 Introduction

A modification was made to CAMRAD.Mod1 to include a tunnel and/or fuselage correction model in the aerodynamic calculations. Actually, the information provided to the code through this modification does not inherently assume a fuselage or tunnel wall; it could be any body introducing a velocity field near the rotor. The correction involves including an additional velocity distribution and an additional wake distortion due to the influence of wind tunnel walls or due to a fuselage body (or due to any object that produces a steady velocity distribution and wake distortion for the rotor in question). The additional velocity distribution is superimposed on the wake induced velocity at the rotor in the circulation iteration (see Figure 1.2), which is used to calculate airloads. The additional wake distortion is added to the wake distortion used to calculate the wake influence coefficients in the trim iteration (see Figure 1.2).

The additional velocity over the disk and additional wake distortion are fixed, user input quantities read from two files for each rotor being used. The first file contains the additional velocity for all radial and azimuthal stations and one additional velocity at the hub. The additional velocity values (read from the file) should be non-dimensionalized by the rotor tip speed. Depending on the user's method of running CAMRAD.Mod1, the input velocity may be a "total additional velocity" or a "delta additional velocity". If the delta form is used for an isolated rotor in a wind tunnel, for example, the mean velocity correction, externally calculated and removed from the velocities in the velocity file, may be included as an alteration to the advance ratio and to the shaft tilt. The velocity file would therefore consist of an incremental velocity distribution at the rotor disk, due to a tunnel and/or fuselage presence, excluding the mean additional velocity. The other method, the total additional velocity method, involves not modifying the flight condition (*i.e.,* the advance ratio and the shaft tilt), but instead, including the mean velocity correction in the additional velocity distribution. Traditionally, the former method is used since, in a typical wind tunnel test of a rotor, a shaft tilt correction is used to offset wind tunnel wall effects.

The second input file for each rotor contains an additional wake geometry distortion. This additional distortion is used in the code as a correction to

the wake geometry to account for the effects of tunnel walls and/or a fuselage body. The additional distortion is input as an incremental distortion at each wake endpoint location. Since the tip vortex wake endpoints are identified by an azimuth index and an age index inside CAMRAD.Mod1, the additional wake distortion vector is also identified, and input, in the same manner. The additional wake distortion vector is calculated externally by multiplying a local additional velocity at the current wake endpoint location by the time needed for the rotor to advance through one time (azimuth) step. Since this provides an incremental distortion at the wake endpoint location, the total additional wake distortion a wake endpoint is calculated internally as the sum of the incremental distortions along a path from the creation time of the vortex to the current age of the vortex.

The wake geometry of a particular tip vortex segment is determined in CAMRAD.Mod1 using either the rigid wake or the free wake method. In the calculation of the influence coefficients, the total additional wake distortion vector, as discussed previously, is added to the current tip vortex position vector of the wake geometry. This vector is then used in the influence coefficients calculation for the particular vortex segment in question. Since the tip vortex position vector is calculated as needed, not stored, the total additional wake distortion vector is calculated as needed also. For this reason, this procedure is duplicated in the wake geometry subroutines (GEOMP1 and GEOMP2) and in the CFD interface BVI wake geometry calculation subroutine (GEOMBVI).

2.9.2 Equations

CAMRAD.Mod1 calculates the position vector for a wake endpoint as needed. When the position vector is calculated, an additional wake distortion vector is also computed. The additional distortion vector is added to the current wake endpoint according to the location of the wake endpoint in the (ψ, ϕ) coordinate system, where ψ is the azimuth at which the endpoint was released from the blade and ϕ is the current age of the endpoint under consideration. Therefore the wake location $\vec{r}_{wake}$ is found as follows:

$$\vec{r}_{wake}(\psi, \phi) = \vec{r}_{wake,before}(\psi, \phi) + \Delta\vec{r}_{wake}(\psi, \phi) \tag{2.27}$$

where $\vec{r}_{wake,before}(\psi, \phi)$ is the vortex endpoint location vector before the tunnel/fuselage correction and $\Delta\vec{r}_{wake}(\psi, \phi)$ is the additional distortion due to the current correction.

At a radial station, r_b, and an azimuth station, ψ, during the calculation of wake induced velocity in subroutines WAKEN1 and WAKEN2, an additional velocity is superimposed on the previously calculated wake induced velocity at that point:

$$\vec{V}_{ind,total}(r_b,\psi) = \vec{V}_{ind}(r_b,\psi) + \Delta\vec{V}(r_b,\psi) \tag{2.28}$$

where $\Delta\vec{V}(r_b,\psi)$ is the additional velocity from the current correction. As discussed earlier, this velocity could be an incremental (*i.e.,* not including the average additional velocity) or a total velocity (*i.e.,* including the average additional velocity), depending on the user's method of running CAMRAD.Mod1.

2.9.3 Code Modifications

A parameter was added to CAMRAD.Mod1 to control usage of the tunnel/fuselage correction model. The parameter OPWFCOR was added to the namelist NLHIRES (discussed in a later chapter) and is saved in the common blocks INTAZ and INTAZ2, for rotor-1 and rotor-2, respectively. If OPWFCOR = 0 the correction model is not used. If OPWFCOR = 1, the 2 input files, discussed earlier, are read by the subroutines INPTR1 and INPTR2 for rotor-1 and rotor-2, respectively. The following code, containing the required input file formats was added to INPTR1 (the same coding applies to INPTR2):

```
      IF (OPWFCOR .EQ. 1) THEN
      DO 799 N = 1,12
799   READ (11,810) DUMMY
      DO 800 J = 1,MPSI
      READ (11,810) DUMMY
      DO 800 K = 1,MPSI*4
800   READ (11,820) DCORR(I,J,K), I = 1,3)
810   FORMAT (A30)
820   FORMAT (13X,3(3X,F12.6))
      DO 830 N = 1,3
830   READ (12,810) DUMMY
      READ (12,860) (VCORRH(I), I = 1,3)
      DO 850 J = 1,MPSI
      READ (12,810) DUMMY
      DO 850 IR = 1,MRA
```

```
850   READ (12,860) (VCORR(I,IR,J) , I = 1,3)
860   FORMAT (19X,3(3XF12.6))
      ELSE
      ENDIF
```

where MPSI is the number of azimuth stations (usually 36), DCORR is the (input) additional wake distortion, VCORRH and VCORR are the additional velocities at the hub and over the rotor disk, respectively. (VCORR is the input additional velocity over the disk.) Note that the unit numbers are 11 and 12 for the distortion and velocity files, respectively. For rotor-2, the above coding is the same in INPTR2, except the unit numbers 11 and 12 are replaced by the unit numbers 21 and 22. The above coding provides the necessary input file formats. Also note that additional wake distortion information must be provided for 4 wake spirals (see the "DO 800 ..." loop). The data for the corrections is stored in the common blocks WFCORR1 and WFCORR2 for rotors 1 and 2, respectively.

In the subroutines WAKEN1 and WAKEN2, the additional velocity is applied by the following additional coding:

```
      IF (OPWFCOR .EQ. 1) THEN
      DO 777 I  = 1,3
      DO 777 KR = 1,MRA
777   VIND(I,KR,L) = VIND(I,KR,L) + VCORR(I,KR,L)
      ELSE
      ENDIF
```

where VIND is the wake induced velocity at the rotor disk.

Since the position vectors of the wake are always calculated as needed, the total additional wake distortion vector is also generated as needed. The total wake geometry is calculated in three places in the low resolution part of CAMRAD.Mod1; once in the influence coefficient calculations (WAKEC1 and WAKEC2), once in the geometry print subroutines (GEOMP1 and GEOMP2), and once in the CFD interface BVI wake location calculation (GEOMBVI). In WAKEC1 and WAKEC2, the following was added:

```
      REAL DSUM(3)
      ...
      IF (OPWFCOR .EQ. 1) THEN
      JDCNT = LM - K1
```

```
      IF (JDCNT .GT. 0) GOTO 665
      JDCNT = JDCNT + MPSI
      GOTO 664
665   CONTINUE
      DO 670 I = 1,3
      DSUM(I) = 0.
      DO 671 KAGE = 1,K1
      DSUM(I) = DSUM(I) + DCORR(I,JDCNT,KAGE)
671   CONTINUE
670   CONTINUE
      DO 666 I = 1,3
      RSECTE(I) = RSECTE(I) + DSUM(I)
666   RTET(I) = RTET(I) + DSUM(I)
      ELSE
      ENDIF
```

where DSUM is the total additional wake distortion, RTET is the position vector of the tip vortex, RSECTE is the position of the secondary tip vortex. In the subroutine GEOMBVI, the same coding is used as above, except the variable K1 is renamed K, and the variable RTET is named RTV.

In GEOMP1 and GEOMP2, code was added to include the additional distortion, and to allow for printing the wake at a resolution higher than 10 $^{\circ}$ azimuth steps. In order to accomplish this, the wake geometry and the distortion are linearly interpolated to a higher resolution by the following code modification (the high resolution modifications will be discussed in a later section):

```
      REAL DCORHI(3),DCORLO(3)
      ...
      IF (OPWFCOR .EQ. 1) THEN
      JDCNTLO = JREFLO - KOUNT
664   IF (JDCNTLO .GT. 0) GOTO 665
      JDCNTLO = JDCNTLO + MPSI
      GOTO 664
665   CONTINUE
      JDCNTHI = JREFHI - K
668   IF (JDCNTHI .GT. 0) GOTO 667
      JDCNTHI = JDCNTHI + MPSI
      GOTO 668
```

```
667   CONTINUE
      ENDIF
      DO 801 I = 1,3
      IF (OPWFCOR .EQ. 1) THEN
      IF (KOUNT .EQ. 0) THEN
      DCORLO(I) = 0.
      ELSE
      DCORLO(I) = 0.
      DO 811 KAGE = 1,KOUNT
      DCORLO(I) = DCORLO(I) + DCORR(I,JDCNTHI,KAGE)
811   CONTINUE
      ENDIF
      DCORHI(I) = 0.
      DO 816 KAGE = 1,K
      DCORHI(I) = DCORHI(I) + DCORR(I,JDCNTHI,KAGE)
816   CONTINUE
      ELSE
      DCORLO(I) = 0.
      DCORHI(I) = 0.
      ENDIF
      RWT(I)  = (RWA(I)+DCORLO(I))*FACT +
                (RWD(I)+DCORHI(I))*FACT2
      RWSO(I) = RWKB(I)*FACT1 + RWKE(I)*FACT2
801   RWSI(I) = RWKC(I)*FACT1 + RWKF(I)*FACT2


      (extensive code added here to account for
      vortex rollup and spinup - see source code for
      details.)
```

2.9.4 Sample Case Discussion

The effect of the tunnel and fuselage corrections on the lift coefficient are shown in Figure 2.14. The body used in this case is the sting/fuselage depicted in Ref. [9]. The tunnel used in this case is the DNW tunnel also depicted in Ref. [9]. Figure 2.14 (a) shows C_l contours without the tunnel/fuselage correction while Figure 2.14 (b) shows contours of the same case, except the tunnel/fuselage correction model is used for a typical rotor wind tunnel "fuselage" in an open wall wind tunnel. Figure 2.15 is the same

information, again plotted radially at several azimuth stations. For this case, the average shaft tilt plus an incremental velocity change method was used. To account for the average shaft tilt correction, the shaft tilt was changed from 5.3 ° aft for the case without the correction applied to 4.233 ° aft for the case with the correction applied. Thus the average shaft tilt portion of the correction was 1.067 ° forward tilt. The average change in advance ratio for this case was negligible ($\mu_{correction} < .001$) and thus was not applied.

2.9.5 Extensions to High Resolution

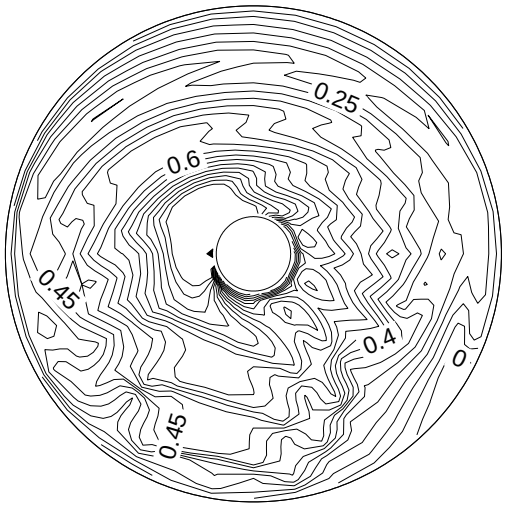
During the wake influence coefficient calculations in the high resolution part of the code, the wake position is interpolated from known quantities. To avoid unnecessary interpolation, the additional velocity over the rotor disk is initially linearly interpolated to the high resolution radial and azimuthal locations. This interpolated value is superimposed on the wake induced velocities in the subroutines VINDCAL1 and VINDCAL2 in the same manner as was done for the low resolution wake induced velocity in WAKEN1 and WAKEN2.

In the high resolution wake influence coefficients calculation, the wake distortion vector at a higher resolution is interpolated from the low resolution information. To avoid unnecessary interpolation, the additional wake distortion is added to the low resolution results before interpolation to the high resolution. This addition is done in the subroutines WKC1INT and WKC1INT in a manner similar to that done in the low resolution subroutines WAKEC1 and WAKEC2.

2.10 Low Resolution Loading Output File

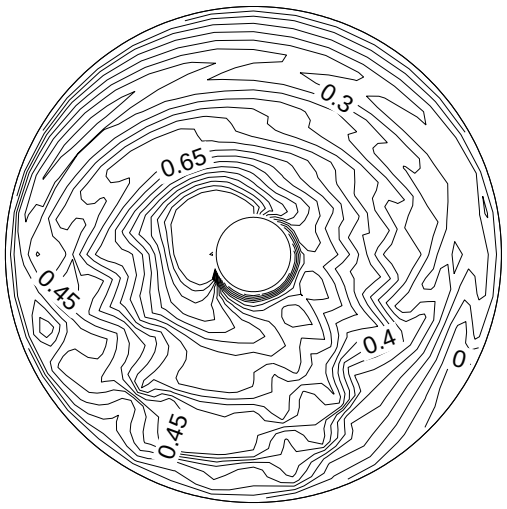
Several subroutines were modified in CAMRAD.Mod1 in order to output information to a file at the end of the trim iteration. Subroutines LOADR1 and LOADR2 were modified to include calls to the new subroutines PRFIL1 and PRFIL2, respectively. For rotor-1, the output information is written to unit number 7 and for rotor-2 the output information is written to unit number 8. The output information, is as follows:

1. Radial stations, $ra(r)$
2. Angles of attack, $\alpha(r, \psi)$
3. Mach number, $M(r, \psi)$



$\psi = 0.0^\circ$

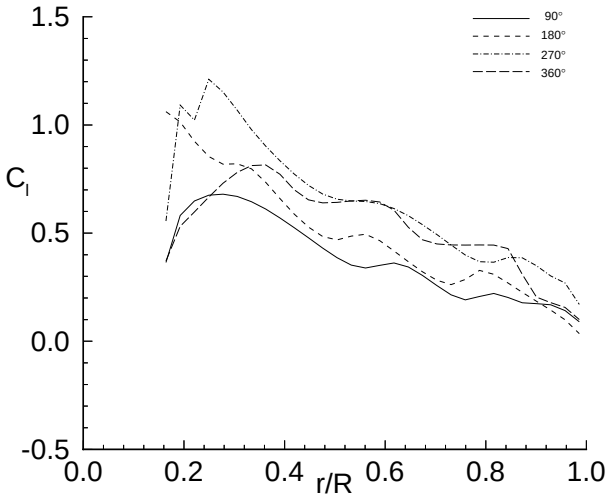
(a) C_l for 10° wake resolution without tunnel/fuselage correction.



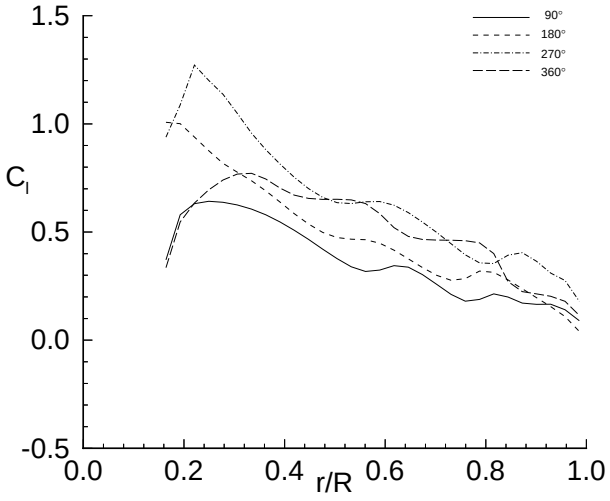
$\psi = 0.0^\circ$

(b) C_l for 10° wake resolution with tunnel/fuselage correction.

Figure 2.14: Contours of local lift coefficient C_l over the rotor disk showing effect of tunnel/fuselage correction.



(a) C_l for 10° resolution $\alpha_{shaft} = 5.3^\circ$ (uncorrected)



(b) C_l for 10° resolution $\alpha_{shaft} = 4.233^\circ$ (corrected)

Figure 2.15: The local lift coefficient as a function of span showing the effect of tunnel/fuselage correction.

4. Tangential velocity, $U_t(r, \psi)$
5. Perpendicular velocity, $U_p(r, \psi)$
6. Radial velocity, $U_r(r, \psi)$
7. Inflow angle, $\phi(r, \psi)$
8. Lift coefficient, $C_l(r, \psi)$
9. Drag coefficient, $C_d(r, \psi)$
10. Moment coefficient, $C_m(r, \psi)$
11. Pitch angle, $\theta(r, \psi)$
12. Tip flap, $\beta(\psi)$
13. Maximum bound circulation, , (ψ)

Except for the $ra(r), \beta(\psi)$, and , (ψ) arrays, the following format is used:

```
     WRITE (XXX,'(A30)') YYY
     DO 25 I = 1,NRAD
25   WRITE (XXX,'(12F11.5)') (ZZZ(I,J) , J = 1,NAZM)
```

where XXX is the unit number 7 or 8, YYY is one of the variable names listed above, ZZZ is the variable associated with the name YYY, I is the radial station index, J is the azimuth station index. The radial station output has the following format:

```
     WRITE (XXX,669)
669  FORMAT (1X,'RADIAL STATIONS')
     WRITE (XXX,670) (RA(I) , I = 1,MRA)
670  FORMAT (1X,10F12.4)
```

and the tip flap and maximum bound circulation have the format:

```
     WRITE (XXX,888)
888  FORMAT (1X,'TIP MOTION - FLAPPING')
     DO 889 J = 1,MPSI
889  WRITE (XXX,890) PSI(J),ZZZ(J)
890  FORMAT (2F11.5)
```

where XXX and J have the same meaning as before, PSI is the azimuth location, and ZZZ is either β or , .

2.11 Wake Geometry and Blade Position Output Files

2.11.1 Introduction

A modification to several subroutines was made in order to write to output files the wake geometry and blade position at an azimuthal resolution that is not necessarily the same resolution as used in the CAMRAD.Mod1 trim procedure. The purpose of printing this information into the output files is for plotting of the wake geometry and blade position information with user supplied plotting programs. For example, if the trim was performed at 10 $^\circ$ azimuth steps, the wake and blade position may now be written out at, say, 5 $^\circ$ steps. The modification involves changes to the subroutines GEOMP1 and GEOMP2. The changes are identical in these subroutines except for the unit numbers and the common blocks used. If the wake geometry print variables in namelist NLLOAD are being used, the wake geometry and blade position information are written to files at the resolution defined by the variable MPSIWGP in the namelist NLHIRES. This variable is the number of azimuth steps per revolution for the wake geometry printout. For example, if MPSIWGP = 72, the wake geometry and blade position would be written to files at a 5 $^\circ$ per azimuth step resolution (*i.e.,* (360 $^\circ$ per revolution)/(72 steps per revolution)).

2.11.2 Code Changes

The tip vortex wake geometry information at the high azimuthal resolution is derived from the low resolution information by linear azimuthal interpolation. The coding added to GEOMP1 to output requested information is as follows:

```
KMAX = MAXO(KFW,KDW,KNW,KRW)
IF (MPSIWGP .LE. 0) GOTO 2
DO 777 J = 1,MPSIWGP
XPSI = FLOAT(J)*360./FLOAT(MPSIWGP)
IZER = 0
XPHI = 0.
CALL FINDRAD (...)
CALL GEOME1  (...)
CALL GEOME1  (...)
PSIRAD = XPSI*PI/180.
```

```
      CALL WAKEB1  (...)
      DO 800 I = 1,3
      RWT(I) = TDUMB(I)
      RWSO(I) = TDUMB(I)
      RWSI(I) = RDUMB(I)

      (extensive coding added to implement rollup
      (and spinup).

      (tunnel/fuselage corrections added here.)

      WRITE (XXX,778) IZER,XPHI,J,XPSI,RWT(1),
     1RWT(2),RWT(3),RSEC(1),RSEC(2),RSEC(3)
      DO 777 K = 1,KMAX
      XPHI = FLOAT(K)*360./FLOAT(MPSI)
      KOUNT = K-1
      IF (ABS(1.-FACT1) .LT. 0.001) KOUNT = K
      CALL GEOME1  (...)
      CALL GEOME1  (...)
      WRITE (XXX,778) IZER,XPHI,J,XPSI,
     1RWT(1),RWT(2),RWT(3),RSEC(1),RSEC(2),RSEC(3)
  777 CONTINUE
  778 FORMAT (2X,I8,F10.2,I8,F10.2,6F12.5)
```

where XXX is the unit number 13 for rotor-1 (GEOMP1) or unit 23 for rotor-2 (GEOMP2), and the subroutine FINDAZ is a new subroutine used to find the linear interpolation factors for a given azimuth location. For use in GEOMP2, the same coding is used, except the calls to GEOME1 are changed to calls to GEOME2, and calls to WAKEB1 are changed to calls to WAKEB2. Actual arguments to the subroutines denoted by "(...)" here, may be found in the source code. The coding added to GEOMP1 to output the blade position information is as follows:

```
      DO 886 JJ = 1,MPSIWGP
      XXPSI = FLOAT(JJ)*360./FLOAT(MPSIWGP)
      XXPSI = XXPSI*PI/180.
      CALL WAKEB1 (...)
      DO 886 I = 1,3
      XROOTB(I) = RDUMB(I)
```

```
      XTIPB(I) = TDUMB(I)
 886  CONTINUE
      DO 888 JJ = 1,MPSIWGP
      XXPSI = FLOAT(JJ)*360./FLOAT(MPSIWGP)
      XXPSI = XXPSI*PI/180.
      CALL WAKEB1 (...)
      DO 888 I = 1,3
      DO 887 II = 1,NRAD
 887  XMIDB(I,II,JJ) = MDUMB(I,II)
 888  CONTINUE
     DO 889 I = 1,3
     DO 889 JJ = 1,MPSIWGP
889  WRITE (YYY,890) XROOTB(I,JJ),(XMID(I,II,JJ),
    1                II=1,NRAD),XTIPB(I,JJ)
890  FORMAT (2X,10F12.5)
```

where YYY is the unit number 14 for rotor-1 (GEOMP1) or unit 24 for rotor-2 (GEOMP2), Again for GEOMP2, the calls to WAKEB1 are replaced by WAKEB2 and the actual arguments may be found in the source code.

2.11.3 Known Caveats

The wake geometry printer plot does not include any of the modifications listed in this section - only the printed output wake geometry file includes these modifications.

2.11.4 Extensions to High Resolution

The high resolution calculations are independent of these wake geometry output modifications. These modifications are simply for the purpose of printing the wake geometry into an output file for plotting.

2.12 Tip Core Size Modifications

2.12.1 Introduction

In the early development of CAMRAD.Mod1, it was determined that a single-sized tip vortex was not adequate to model certain effects such as blade vortex interactions. Several of the early modifications are described in this section. First, the original single tip vortex model was modified to allow core

size changes as a function of wake age. This single vortex model produced a relatively smooth velocity profile radially out from the center of the vortex. It was seen in several experimental investigations that the vortex velocity profile was, in some cases, quite different from the well known "Scully-type" vortex profile. As an attempt to study the effect of changing the velocity profiles, and to have these profiles related to the loading on the blade, a dual core model was implemented. The dual core model consisted of two concentric tip vortices (see Figure 2.16) such that each contained a portion of the maximum bound circulation. More recently, a model of the vortex rollup process was implemented that supersedes this section. However, if the rollup model is not being used, the models discussed here may be used. This section is mainly for historical value and was written to document the single tip vortex core variable size model and the dual core model in CAMRAD.Mod1 The vortex rollup model, which is described in a subsequent section, is the recommended model.

2.12.2 Single Core Modifications

The single tip vortex core model in CAMRAD.Mod1 was modified to allow a variable core size depending on the age of the vortex segment. First, the single core model is chosen by selecting the variable OPDCORE = 0 in namelist NLHIRES. Then the options for the variable tip core size are chosen via the variable OPTVCOR, also in namelist NLHIRES. If OPTVCOR = 0, then the original single core model is retained. If OPTVCOR = 11, then a step function is applied to the core size. The core size, in this case, starts at the size defined by the input variable, CORE(1), in namelist NLWAKE. The core size increases to the value RCORINC (in namelist NLHIRES) at the vortex age PHIINC (also in namelist NLHIRES). If a core size function other than a constant or a step function is desired, OPTVCOR = 12 may be chosen. This choice applies a tenth order polynomial in wake age to the core size. Any agewise core size variation function may be approximated by this truncated series. A summary of single core size choices is listed in namelist NLHIRES in Chapter 5.

2.12.3 Sample Case Discussion

As an example of the modification, the lift coefficient from the 10 ° case from previous sections is shown in Figure 2.17, along with the same case including this modification. Figure 2.17 (a) is the same 10 ° azimuth case

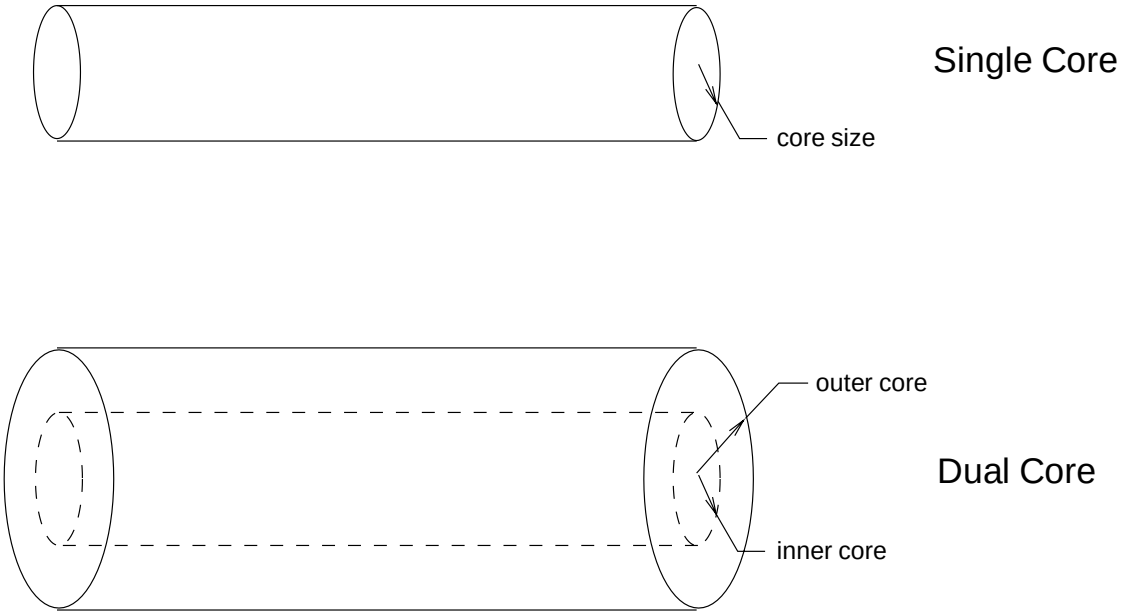


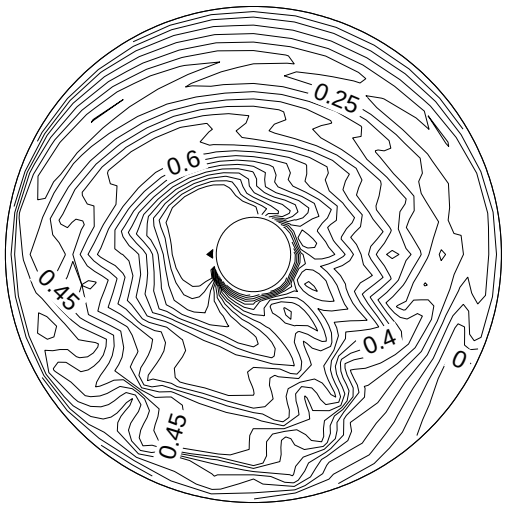
Figure 2.16: Definition of single and dual core models implemented in CAM-RAD.Mod1

with a constant core size as before. Figure 2.17 (b) is the same case, but the single core step function is applied. Figure 2.18 is the same information plotted radially at several azimuth stations. For this case, the step increase in core size is made after the equivalent of one revolution of wake age. That is, the step function is at a wake age of 360 $^\circ$ (PHIINC = 360 $^\circ$), and the core radius was increased from CORE(1) = .0212R to RCORINC = .091R. The loading for the large core case is seen to be much smoother (Figures 2.17 (a) and 2.17 (b)) in the rear of the disk than in the small core case, as expected. Even though the core size increase in this example is unrealistic, the effect of the modification is shown well. The core size may, in reality, vary as a more general function of wake age. In such a case, the function may be fitted with the tenth order polynomial and the case may be run with OPTVCOR = 12.

2.12.4 Dual Core Modifications

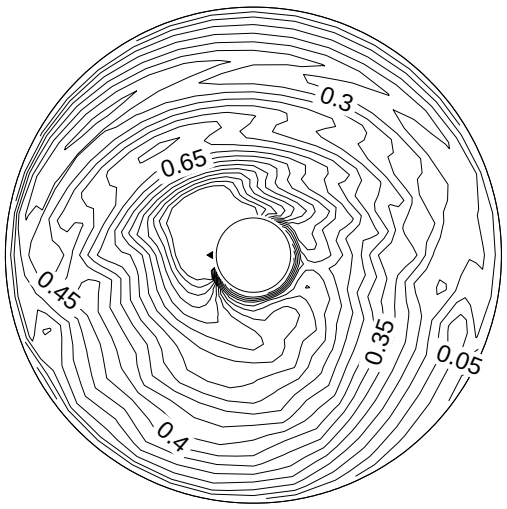
A modification was added to CAMRAD.Mod1 to include a "dual core tip vortex" model. This model essentially alters the velocity profile of a vortex core by superimposing two vortex cores of different sizes (*i.e.,* inner and outer core) and strengths. In CAMRAD.Mod1, a "core model factor" is defined which multiplies the influence coefficients for particular vortex segments. This factor can be used as a tool to simulate velocity distributions in viscous vortex cores. Several choices, listed in namelist NLHIRES (see Chapter 5), may be made when using the dual core model. If the variable OPDCORE = 1, then the dual core model will be used in the code. If this choice is made, several other parameters in namelist NLHIRES must be set by the user. For example, the variable OPCORAC specifies the manner in which the inner core size is calculated. If OPCORAC = 0, the inner core and outer core sizes are user specified constants. If OPCORAC = 1, the inner core size is determined from internal calculations based on the angle of attack of the last aerodynamic collocation point along with user specified constants.

Similar to the single core modifications, there is an option to vary the core size with wake age for the dual core model. The core size variation function choices, for the dual core option, does not include a step function option, but does include a tenth order polynomial in wake age for both the inner and outer vortex cores. The parameters in this core size function determine the usage. There is not a variable, as in the single core model, used to turn on or off the core size function. Of the parameters listed below,



$\psi = 0.0^\circ$

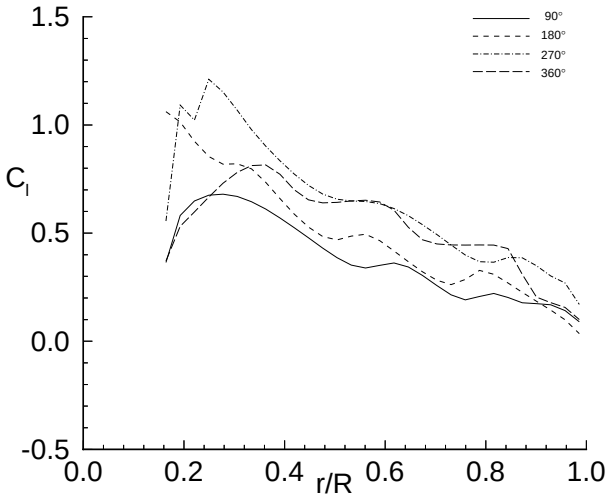
(a) C_l for 10° wake resolution with constant size single core.



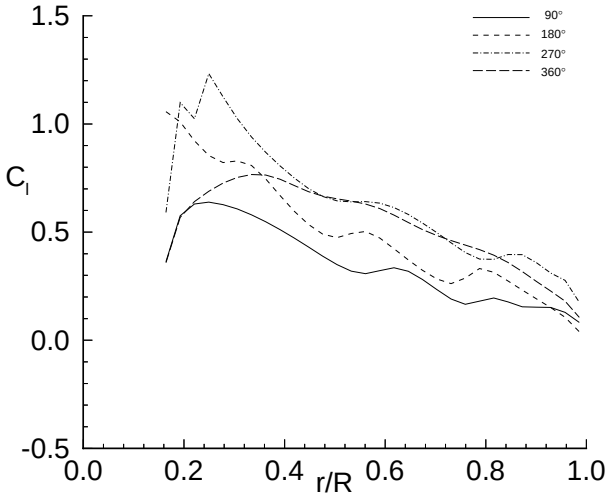
$\psi = 0.0^\circ$

(b) C_l for 10° wake resolution with single core step function.

Figure 2.17: Contours of local lift coefficient C_l over the rotor disk showing effect of single core step function.



(a) C_l for 10 ° wake resolution with constant size single core.



(b) C_l for 10 ° wake resolution with single core step function.

Figure 2.18: The local lift coefficient as a function of span showing the effect of wake model specified with single core step function.

if all are set to zero, the result is that no dual core agewise size changes occur. These parameters are set to values other than zero to produce the desired core size changes with wake age. The functions are as follows:

$$r_i = r_{inner} + d_{inner,0} + \sum_{n=1}^{N1} d_{inner,n}\phi^n \tag{2.29}$$

$$r_o = r_{outer} + d_{outer,0} + \sum_{n=1}^{N2} d_{outer,n}\phi^n \tag{2.30}$$

where $d_{inner,0}$ is the variable DCCORFA0 in NLHIRES, $d_{inner,n}$ is the set of variables DCCORFA(n) in NLHIRES, $d_{outer,0}$ is the variable DCCORFB0 in NLHIRES, $d_{outer,n}$ is the set of variables DCCORFB(n) in NLHIRES, N1 is the variable NDCCOFA in NLHIRES, N2 is the variable NDCCOFB in NLHIRES, r_{inner} is the current inner core radius, and r_{outer} is the current outer core radius.

If the dual core model is used, the strength if the vortex must be split between the two such that the total strength remains the same. To accomplish this, the inner core influence coefficient is multiplied by a factor, β, and the outer core is multiplied by $(1 - \beta)$. The value of β implemented is proportional to the blade chord at the last aerodynamic collocation point on the blade, to the angle of attack at the last aerodynamic collocation point on the blade, and to the velocity at the last aerodynamic collocation point on the blade. The constant of proportionality is a user specified constant. The calculations of β are done internally at the vortex leading and trailing edges. Also, the value of β is non-dimensionalized by the maximum bound circulation associated with the current vortex segment. The value of β is thus:

$$\beta = \frac{(\gamma \pi c U \alpha)}{, _{max}} \tag{2.31}$$

where γ is the user specified constant GAMACST in namelist NLHIRES, c is the blade chord at the last aerodynamic collocation point, U is the velocity at the last aerodynamic collocation point, α is the angle of attack at the last aerodynamic collocation point, and $, _{max}$ is the maximum bound circulation at the current azimuth location.

2.12.5 Sample Case Discussion

As an example of this modification, Figure 2.19 shows the lift coefficient contours of the standard 10 ° case along with the same case using the dual core model. In this case, a constant inner and outer core size is used (OPCORAC = 0). No core size "age function" was used and a value of 0.1 was used for GAMACST. Figure 2.20 shows the same information, plotted at several azimuth angles.

2.12.6 Code Modifications

Code changes for this modification are made in subroutines WAKEC1 and WAKEC2. At each occurrence of the single core influence coefficient subroutine VTXL, there is now an IF statement involving the variable OPDCORE. If OPDCORE = 0, the single core model is used as described earlier. If OPDCORE = 1, the dual core model is used as described previously. The dual core model uses a new subroutine VTXL2 to calculate the dual core influence coefficients. VTXL2 is identical to the VTXL except the core size effect on the influence coefficients is calculated in the calling routine instead of inside VTXL2.

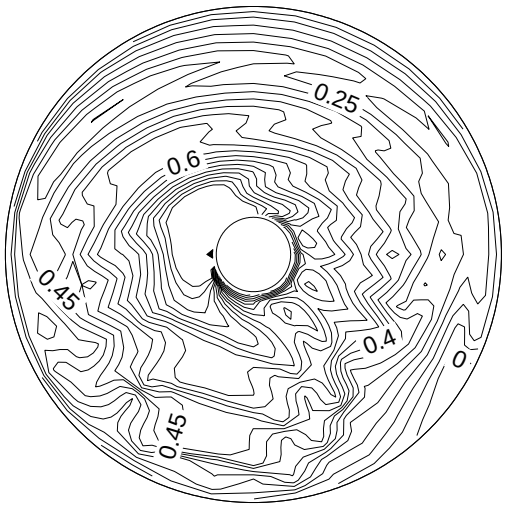
2.12.7 Known Caveats

1. The dual core model and the rollup model should be mutually exclusive, but the code does not check to see if only one of these is being used.

2. In the original version of CAMRAD, the wake influence coefficient calculations assumed a constant vortex core radius. Since the core sizes in this section are variable, strictly speaking, the variation of core radius should be accounted for in the integrals that are used to derive the influence coefficients. However, as an approximation, these effects are neglected and are expected to be small.

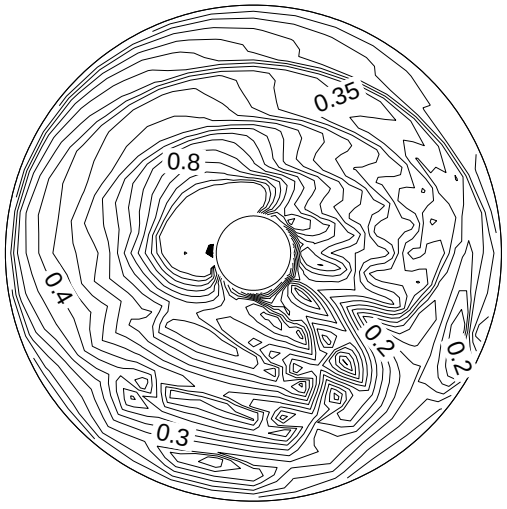
2.12.8 Extensions to High Resolution

The same modifications are included in the HIRES portion of the code. For the single core model, the step function or the tenth order polynomial may be applied either in the low resolution only, or in the low and high resolution portions, depending on the value of OPTVCOR. The core size



$\psi = 0.0^\circ$

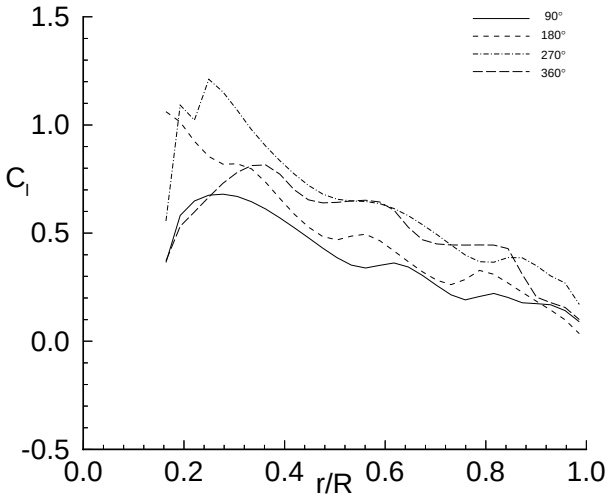
(a) C_l for 10° wake resolution with constant size single core.



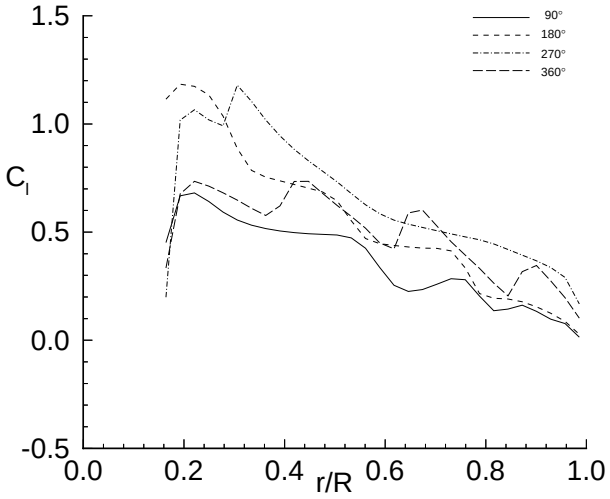
$\psi = 0.0^\circ$

(b) C_l for 10° wake resolution with dual core model.

Figure 2.19: Contours of local lift coefficient C_l over the rotor disk showing effect of dual core model used in free wake computation.



(a) C_l for 10 ° wake resolution with constant size single core.



(b) C_l for 10 ° wake resolution with dual core model.

Figure 2.20: The local lift coefficient as a function of span showing the effect of dual core model used in free wake computation.

expansion functions applied to the dual core model are always in effect in both low and high resolution.

2.13 Modification for Tip Vortex Bursting

2.13.1 Introduction

A modification was made to the low resolution part of CAMRAD.Mod1 to include vortex bursting for the single tip vortex core model. When a vortex is sufficiently close to a blade, bursting may occur and the vortex size and/or strength (circulation) altered. In CAMRAD.Mod1, for bursting to occur, two criteria must be met: (1) a vortex must either cross the blade, or have an endpoint sufficiently close to the blade (*i.e.,* within a user specified azimuthal tolerance, PSITOL), and (2) the distance between the blade and the vortex at the blade-vortex crossing point must be less than or equal to a user specified tolerance, ZTOL ($\Delta z/R$). To locate the blade-vortex crossing point it is assumed that the lead-lag displacement of the blade is negligible and that the blade has a no sweep. With these assumptions, the blade is modeled as a straight line at each azimuth angle. This line model of the blade may then be compared to the location of all wake segments to determine which vortex segments cross, or are within an azimuthal tolerance of, a blade (see Figure 2.21).

The procedure for the bursting calculation starts with the initialization of the burst vortex core size to the input core size. Since the burst vortex strength is calculated as a fraction of the maximum bound circulation, the fraction is initialized to unity so that the full strength vortex is used at the beginning of the calculation. A blade test parameter is initialized to zero. This parameter, which monitors which blade has burst which vortex segments, is used in subsequent wake-trim iterations to avoid re-bursting by the same blade-vortex encounter. Initializing this parameter to zero means "no blade has burst any vortex segment".

At each azimuth step in the influence coefficients calculation, the burst core size, strength fraction, and blade test parameter are initialized to the appropriate value determined by the azimuth angle and wake age. For example, for an azimuth angle of $\psi = 100^\circ$ and a wake age of $\phi = 100^\circ$, the burst core size is initialized to the value at $\psi = 90^\circ$ with a wake age of $\phi = 90^\circ$. This initialization scheme convects the burst parameters with the burst vortex as the burst vortex is convected downstream. At the current azimuth location, all vortex segments of the reference blade are tested to

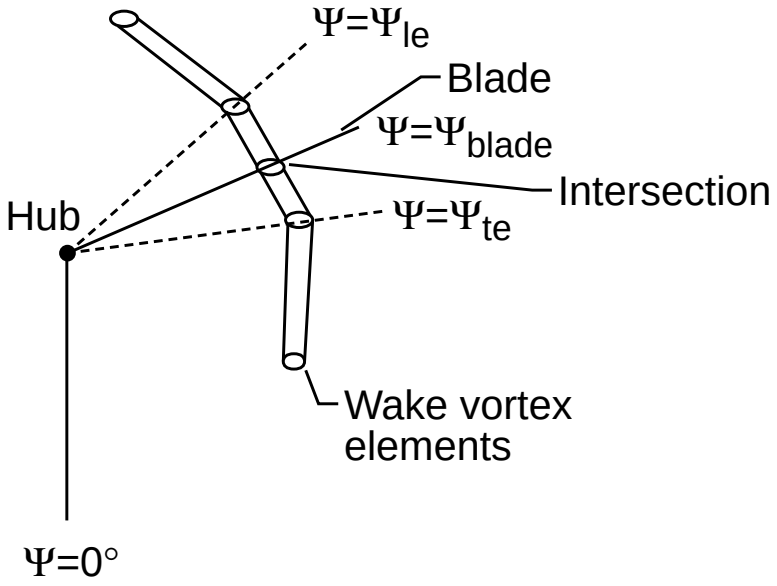


Figure 2.21: Schematic of a blade-vortex intersection.

determine if there is a vortex crossing of any blade on the rotor or if the vortex endpoints are within the azimuthal tolerance PSITOL. Of the following criteria (Equations 2.32, 2.33, and 2.34), if the first criterion is met, there is a blade-vortex crossing; if the second or third criterion is met, the vortex is within the azimuthal tolerance PSITOL of the blade:

$$(\psi_{blade} - \psi_{le})(\psi_{blade} - \psi_{te}) \leq 0. \tag{2.32}$$

$$|\psi_{blade} - \psi_{le}| \leq PSITOL \tag{2.33}$$

$$|\psi_{blade} - \psi_{te}| \leq PSITOL \tag{2.34}$$

where ψ_{blade}, ψ_{le}, and ψ_{te} are the azimuth angles of the line model of the blade, of the leading edge of the vortex , and of the trailing edge of the vortex, respectively (see Figure 2.21).

If a blade azimuth-vortex crossing is detected, the intersection point is located by first calculating the radial location of the intersection along the blade. If the radial location of the "crossing" is less than the innermost radial station on the blade or if it is outboard of the blade tip, then the "crossing" is ignored since there is no blade segment involved. With the

radial station of the intersection known, the vertical coordinate of the the blade and of the wake is found at the intersection by linear interpolation between known blade and wake coordinates. The difference in these vertical coordinates is the blade-vortex intersection "miss-distance". If the miss-distance is greater than the user specified tolerance, ZTOL, the intersection does not burst the segment. If it is less than, or equal to the vertical tolerance, the vortex is burst. The bursting changes the vortex core size and the vortex strength. Also, the blade test parameter is set equal to unity so that in a later wake-trim iteration, the same blade-vortex intersection does not generate an additional burst of the same vortex.

If the vortex segment is within the specified azimuth tolerance, the "intersection" point is calculated as the closest point on the blade to the closer of the leading edge or trailing edge of the vortex segment. Again, if the "intersection" point is inboard of the innermost radial station, or outboard of the tip, the intersection is ignored. Otherwise, the blade height is determined by linear interpolation of known blade information, and the vortex height is assumed to be the vertical coordinate of the endpoint that is involved in the "intersection". As before, the "miss-distance" is the difference in the blade z-coordinate and the vortex z-coordinate.

Currently, a simple bursting model is used. For a bursting event, the core size and strength are multiplied by the factors, CORMULT and CIRMULT, respectively. Then the burst core size and strength are saved. As the calculation proceeds, the same vortex may be burst again by another blade. However, the same vortex may not be burst again by the same burst event during a subsequent wake-trim iteration.

2.13.2 Sample Case Discussion

As an example of this modification, the 10 ° case without bursting is compared to the case with bursting. This example is intended to show the effect of no bursting compared to the case were all vortices have been burst. This is not the intended usage of the model, but illustrates that bursting may have a significant effect on the predictions. Figure 2.22 shows a comparison of lift coefficient contours for these two cases. The case in Figure 2.22 (b) uses the bursting model (OPBURST = 1) with only the elements "crossing" a blade being burst (PSITOL $\leq$ 0.0). The vertical tolerance in this case is set to a relatively large value (ZTOL = 0.1) in order to dramatically show the effect of the bursting model. This value effectively bursts all wake elements on the rotor disk. For this forward flight case, all wake elements

are within a 0.1R vertical distance of the blades. Figure 2.23 illustrates the same information, but plotted as a function of radius for several azimuth angles.

2.13.3 Code Modifications

Changes were made to INPTW1 and INPTW2 to include a namelist read of the new namelist NLBURST. This namelist is read after NLWAKE for each rotor. (The namelist NLBURST parameters are listed in Chapter 5.) Also, included in INPTW1 and INPTW2 is the initialization of the arrays COREBLO and CIRCBLO. These are the burst core size and strength fraction arrays, respectively. COREBLO is initialized to the input core size. CIRCBLO is initialized to unity. New common blocks, BURST and BURST2, for rotor-1 and rotor-2, respectively, were also added. Changes were made to subroutines WAKEC1 and WAKEC2 to include vortex bursting. The common blocks BURST and BURST2 were added to these subroutines. Loops were added to calculate the effects of bursting. One of these loops calculates and stores the blade position at all radial and azimuthal stations. Another loop initializes the arrays COREBLO, CIRCBLO, and INBSTLO at each new reference blade azimuth location as described in above. INBSTLO is the blade test parameter. It is an array in the common blocks BURST and BURST2 used to prevent bursting of a vortex that was burst by the same burst event during a previous free wake-trim iteration. A loop was added inside the wake age loop of the influence coefficients calculation to determine if a bursting event occurs and if so, to set COREBLO, CIRCBLO, and INBSTLO to appropriate values.

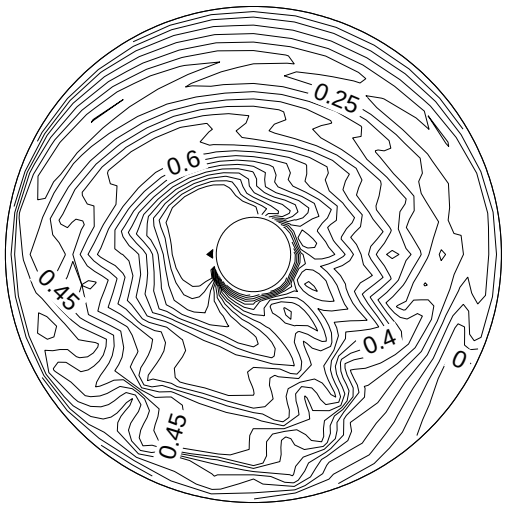
2.13.4 Known Caveats

1. The circulation burst model (as opposed to the core size burst model) is not operational in CAMRAD.Mod1.

2. The burst model (circulation burst or core size burst) has not been exercised.

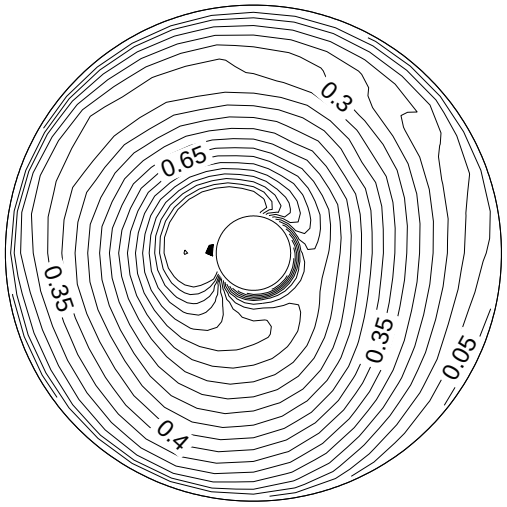
3. The burst model is implemented for the single tip vortex core model only.

4. The burst model is not implemented for the rollup model.



$\psi = 0.0^\circ$

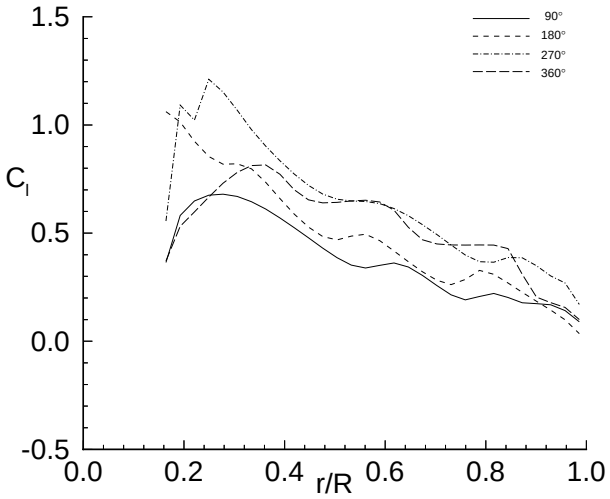
(a) C_l for 10° wake resolution without bursting model.



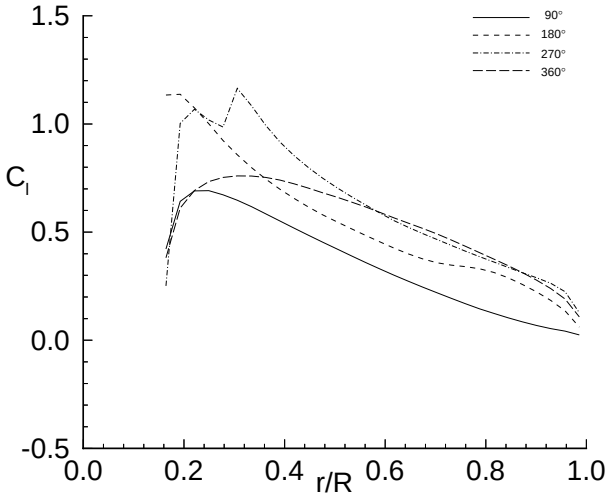
$\psi = 0.0^\circ$

(b) C_l for 10° wake resolution with bursting model.

Figure 2.22: Contours of local lift coefficient C_l over the rotor disk showing effect of the wake bursting model.



(a) C_l for 10 ° wake resolution without bursting model.



(b) C_l for 10 ° wake resolution with bursting model.

Figure 2.23: The local lift coefficient as a function of span showing the effect of the wake bursting model.

5. The influence coefficient calculations in the original version of CAMRAD were derived for a constant vortex core size. Here, the core size changes with wake age and azimuth. The influence coefficients used here do not include the effect of a variable core size with wake age or azimuth angle.

2.13.5 Extensions to High Resolution

The same modifications above apply to the high resolution portion of the code. The changes needed for the high resolution application are that the arrays COREBLO, CIRCBLO, and INBSTLO are larger and are named COREBHI, CIRCBHI, and INBSTHI. The tolerances are the same as in the low resolution portion. The blade test parameter INBSTHI insures that subsequent interactions in the high resolution far wake do not re-burst the same vortex segment by the same blade.

2.14 Modifications to Use Input Blade Motion

2.14.1 Introduction

A technique was developed to allow the user to input elastic blade motions directly into CAMRAD.Mod1 and use the input blade motion values to calculate the resulting aerodynamics. These elastic blade motions may be obtained from other analyses or measurements. The current modification was tested using measured blade motions, but predicted motions could have been used just as well. If the input blade motions are used, the internal calculation of blade motions via an iterative harmonic analysis of the rotor equations of motion, is overridden. Several new subroutines were included to allow measured blade motions to be used in CAMRAD.Mod1.

If using a cantilever blade model (HINGE = 1 in namelist NLRTR), MOTNIN_FL and MOTNIN_P read the flap/lag elastic motion and the pitch motion, respectively, from the files provided by the namelist variables FLAPFILE, LAGFILE, and PITCHFILE (from the new namelist NLMEAS discussed later) which contain the elastic blade motion in a TECPLOT (see Ref. [12]) format. If using a hinged rotor (HINGE = 0 in namelist NLRTR), the same procedure is followed as above except MOTNIN_FLA is used instead of MOTNIN_FL. In addition, the variables FLAP0 and LAG0 define the measured mean flap and lag angles for the hinged blade for use in MOTNIN_FLA. The hinged blade motion option has not at present time

been tested. If the option HINGE = 2 (articulated blade model) is chosen from namelist NLRTR, an error message is printed from subroutine RAMF and the code stops. This is because the articulated blade option (HINGE = 2) contains no elastic blade motion - only rigid flap and lag.

All three files, FLAPFILE, LAGFILE, and PITCHFILE, contain measured elastic motion at each measured radial station, each in a separate TECPLOT "zone" (see below). The number of radial and azimuth stations read are determined by the variables RIN and MPSIIN, respectively, in the namelist NLMEAS. For example, the file FLAPFILE contains the lines:

```
zone
index, flap
index, flap
index, flap
   .
   .
   .
```

where "index" is a number associated with the azimuth location of the elastic flap value, "flap". (The value of "index" is not actually used by the analysis - the input parameter MPSIIN is used to determine the number of lines in the file and thus the azimuth location of the "flap" value.)

Each zone is a list of the measured values (flap, lag, or pitch) at the current radial station and contains the same number of lines that corresponds to the number of measured azimuth stations. The flap and lag values MUST be input in centimeters. The pitch values MUST be input in degrees. For example, the file FLAPFILE contains RIN zones and MPSIIN entries in each zone. The first zone in FLAPFILE contains the azimuthal time history of the elastic flap for the first measured radial station; the second zone contains the azimuthal time history at the second measured radial station, *etc.* This zone convention is repeated for all the measured radial stations. This same convention is followed in all three input files, *i.e.,* the FLAPFILE, LAGFILE, PITCHFILE files. (Note on interpolation vs. extrapolation: To avoid extrapolation, the first zone in each file should be a "dummy" zone containing all zeros for a radial station inboard of the innermost radial station used in the CAMRAD.Mod1 analysis. Likewise, the last zone could be a zone containing information at r/R = 1.)

The subroutine IUNI is an interpolation routine used to interpolate data onto a desired grid, and SFTMOD is a modified version of the WOPWOP Fourier Transform subroutine, SFT.

The modification is implemented as an option controlled by the variable IMODEIN in the namelist NLMEAS. If IMODEIN = 0, the normal CAMRAD.Mod1 modal analysis blade motion prediction is used. If IMODEIN = 1, the input elastic motions are used instead. The maximum dimensions for input elastic motions are set to 50 radial stations and 2049 azimuth stations.

2.14.2 Method for Bending Modes

In order to use measured blade motions in CAMRAD.Mod1, the motion must be cast into a suitable form. Since CAMRAD.Mod1 uses a modal analysis to describe the mode shapes of the rotor blade, it is required to cast the measured deflections in the form of harmonics of modal amplitudes. Since the flap/lag mode shapes are decoupled from the torsion mode shapes in CAMRAD.Mod1, it is possible to split the problem into 2 parts: (1) the flap/lag deflection analysis, and (2) the torsion analysis. First, the flap/lag analysis will be discussed.

The modal equation for the flap/lag deflection is as follows:

$$\vec{z}(r,\psi) = \sum_{k=1}^{nbm} \vec{\eta}_k(r) q_k(\psi) \tag{2.35}$$

where,
$\vec{z}(r,\psi)$ = measured deflection (flap,lag)
$\vec{\eta}_k(r)$ = bending mode shape of the kth mode (flap,lag)
$q_k(\psi)$ = modal amplitude of the kth mode
nbm = number of bending modes

The unknown in Equation 2.35 is $q_k(\psi)$. To solve for $q_k(\psi)$, a system of nbm equations is generated at each azimuth location, ψ, by multiplying (dot product) the modal equation above by a mode shape, $\vec{\eta}_m(r)$, and integrating radially over the span to eliminate the radial dependence. The result is a $nbm \times nbm$ linear system at a given azimuth:

$$\int_0^1 \vec{\eta}_m(r) \cdot \vec{z}(r,\psi) dr = \sum_{k=1}^{nbm} q_k(\psi) \int_0^1 \vec{\eta}_m(r) \cdot \vec{\eta}_k(r) dr \tag{2.36}$$

where $1 \leq m \leq nbm$. The integration has been moved inside the summation, since both are linear operators and $q_k(\psi)$ has been moved outside the integration, since it is not a function of radius. In matrix form, Equation 2.36 becomes:

$$[A_{mk}]\{q_k(\psi)\} = \{B_m\} \tag{2.37}$$

where each element of $A_{mk} = \int_0^1 \vec{\eta}_m(r) \cdot \vec{\eta}_k(r) dr$, and each element of $B_m = \int_0^1 \vec{\eta}_m(r) \cdot \vec{z}(r,\psi) dr$. The solution of Equation 2.37 is as follows:

$$\{q_k(\psi)\} = [A_{mk}]^{-1}\{B_m\} \tag{2.38}$$

After this step, the modal amplitude that produces the measured deflection, at the current azimuth, is known. This procedure is repeated for MPSI azimuth stations. Since CAMRAD.Mod1 uses the complex Fourier coefficients of the modal amplitudes instead of the modal amplitudes directly, the modal amplitudes just calculated must be decomposed into its Fourier components. The modal amplitude for the k-th bending mode may be represented as a series summed over the number of harmonics used:

$$q_k(\psi) = \sum_{n=0}^{mharm} (\beta_{nc}^{(k)} \cos(n\psi) + \beta_{ns}^{(k)} \sin(n\psi)) \tag{2.39}$$

and the coefficients of the series $\beta_{nc}^{(k)}$ and $\beta_{ns}^{(k)}$ are related to the complex coefficients by:

$$\beta_n^{(k)} = \frac{\beta_{nc}^{(k)} - i\beta_{ns}^{(k)}}{2} \tag{2.40}$$

2.14.3 Method for Torsion Modes

The elastic torsion motion is analyzed in a fashion similar to the bending motion analysis in the previous section. However, the torsion motion analysis is a scalar operation rather than a vector operation. The modal equation for the torsion motion is:

$$\theta(r,\psi) = \sum_{k=1}^{ntm} \zeta_k(r) p_k(\psi) \tag{2.41}$$

where,

$\theta(r,\psi)$ = measured torsion deflection
$\zeta_k(r)$ = torsion mode shape of the kth mode
$p_k(\psi)$ = modal amplitude of the kth mode
ntm = number of torsion modes

The solution for $p_k(\psi)$ follows the solution for $q_k(\psi)$ in the previous section, except the modal equation is multiplied by the mode shapes, $\zeta_m(r)$, rather that taking a dot product. The resulting equations are:

$$\int_0^1 \zeta_m(r)\theta(r,\psi)dr = \sum_{k=1}^{ntm} p_k(\psi) \int_0^1 \zeta_m(r)\zeta_k(r)dr \tag{2.42}$$

where $1 \leq m \leq ntm$. The matrix form of which is:

$$[A_{mk}]\{p_k(\psi)\} = \{B_m\} \tag{2.43}$$

The solution of which is:

$$\{p_k(\psi)\} = [A_{mk}]^{-1}\{B_m\} \tag{2.44}$$

The series for the modal amplitudes is:

$$p_k(\psi) = \sum_{n=0}^{mharm} (\theta_{nc}^{(k)} \cos(n\psi) + \theta_{ns}^{(k)} \sin(n\psi)) \tag{2.45}$$

The complex coefficients of which are:

$$\theta_n^{(k)} = \frac{\theta_{nc}^{(k)} - i\theta_{ns}^{(k)}}{2} \tag{2.46}$$

2.14.4 Code Modifications

A new namelist, NLMEAS, was added for rotor-1 only. (That is, the input blade motion modification is currently available only for rotor-1 in CAMRAD.Mod1.) The subroutine INPTN was modified such that if the namelist NLWAKE is read for rotor-1 (OPREAD(2) = 1 in NLTRIM), the subroutine INPTM1 is also called. This new subroutine reads the namelist NLMEAS. The variables in namelist NLMEAS are found in Chapter 5.

The new subroutines, MOTNIN_FL (or MOTNIN_FLA) and MOTNIN_P, read the flap, lag, and torsion measured motion, interpolate the motion onto the grid used in CAMRAD.Mod1, generate a linear system of equations at each azimuth, solve the linear system at each azimuth, Fourier analyze the modal amplitudes, and calculate the complex coefficients by storing the real Fourier coefficients in the appropriate complex arrays.

The call to subroutine INRTM1 in subroutine RAMF was changed to "CALL INRTM1 (IMODEIN)" and the call to MOTNR1 was changed to

"CALL MOTNR1 (IMODEIN)". The subroutine INRTM1 was changed such that if IMODEIN = 1, the motion for bending and torsion is set to the previously calculated complex Fourier coefficients. The line:

```
    BETA(JN,JROW)=B
```

was changed to:

```
    IF (IMODEIN .EQ. 1) THEN
    BETA(JN,JROW) = BETA(JN,JROW)
    ELSE
    BETA(JN,JROW)=B
    ENDIF
```

and the line:

```
    THETA(JN,JROW)=B
```

was changed to:

```
    IF (IMODEIN .EQ. 1) THEN
    THETA(JN,JROW) = THETA(JN,JROW)
    ELSE
    THETA(JN,JROW)=B
    ENDIF
```

The subroutine INRTM1 was changed such that if IMODEIN = 1, the motion for bending and torsion is not updated by the internal CAMRAD.Mod1 motion analysis. The line:

```
42  BETA(N1,I) = BETA(N1,I) + DEL*KH
```

was changed to:

```
    IF (IMODEIN .EQ. 1) THEN
    BETA(N1,I) = BETA(N1,I)
    ELSE
    BETA(N1,I) = BETA(N1,I) + DEL*KH
    ENDIF
42  CONTINUE
```

and the line:

```
44  THETA(N1,I) = THETA(N1,I) + DEL*KH
```

was changed to:

```
    IF (IMODEIN .EQ. 1) THEN
    THETA(N1,I) = THETA(N1,I)
    ELSE
    THETA(N1,I) = THETA(N1,I) + DEL*KH
    ENDIF
44  CONTINUE
```

2.14.5 Extensions to High Resolution

The HIRES portion of the code interpolates the needed motion from known low resolution information. Thus, there are no additional modifications required to the method in order for the high resolution (HIRES) portion of the code to work properly.

2.15 Modifications to Use Input Normal Force Coefficients

2.15.1 Introduction

In a manner similar to the user-provided blade motion input, a modification has been made to allow the user to input a file containing normal force coefficients, C_n, into CAMRAD.Mod1. These may be from another analysis or from measurements. Since measured values of C_n at a blade section are normally calculated from an integration of measured surface pressures around the section, and since accurate dynamic drag coefficients and moment coefficients are normally not available from the pressure data, in this modification it is assumed that the lift coefficient, C_l, is approximated sufficiently by C_n. It is assumed that the drag coefficient, C_d, and the moment coefficient, C_m, are sufficiently determined by the normal CAMRAD.Mod1 airfoil table "look-up" procedure. With these assumptions, the user-input C_n's are interpolated to the resolution needed by the low resolution part of the CAMRAD.Mod1 analysis. These values are then used to replace the internally calculated values of C_l.

The input file for C_n, called CNFILE in namelist NLMEAS, is the same format as described for the FLAPFILE, LAGFILE, and PITCHFILE files. The use of this modification is controlled by the variable IAEROIN in namelist NLMEAS. If IAEROIN = 0 the C_n input file, CNFILE, is not used. If IAEROIN = 1 the C_n input file, CNFILE, is used. The maximum dimensions for input C_n's are set to 50 radial stations and 2049 azimuth stations.

2.15.2 Code Modifications

A new namelist, NLMEAS, was added for rotor-1 only. (That is, this modification is available only for rotor-1 in CAMRAD.Mod1.) The variables in namelist NLMEAS are found in Chapter 5.

If IAEROIN = 1, the subroutine INPTM1 calls the new subroutine MSAERO1 to read and interpolate the CNFILE data. Then the interpolated data is used in either AEROF1 or AERBED1 (depending on the aerodynamic model chosen by the variable OPBED in namelist NLBED for rotor-1). The common blocks AEROMS and AEROIN were added to both of these subroutines.

2.15.3 Known Caveats

This modification has not been exercised nor has it been tested thoroughly.

2.15.4 Extensions to High Resolution

This method has not been implemented in the high resolution calculations to date.

2.16 Indicial Aerodynamics in Low Resolution CAMRAD.Mod1

2.16.1 Introduction

Chapter 4 discusses a computer program called the Indicial Post-Processor (IPP), which is used as one of two methods to process the high resolution information from HIRES (Chapter 3), in determining the rotor blade loading. To study the effects of the indicial aerodynamics models on

the rotor trim solution, a low resolution indicial aerodynamics model was implemented in CAMRAD.Mod1. Since a modification must fit into the overall scheme of CAMRAD.Mod1, it was not possible to directly use the Indicial Post-Processor code for the low resolution implementation. Also, the Indicial Post-Processor was developed and is tuned for high resolution solutions. It has several features that would be inconsistent for use with a low resolution solution. The low resolution implementation of the indicial aerodynamics is the topic of this section.

For simplicity, Figure 2.24 schematically shows an abbreviated flow chart of the original CAMRAD trim loop. The outer rectangle represents the subroutine, TRIM. The three inner rectangles represent the three stages involved in a typical trim solution procedure. These three stages are the uniform inflow stage, the rigid wake stage, and the free wake stage. At each of these stages, a trimmed rotor solution is obtained and the trimmed solution is used to initialize the following stage (if one exists). The subroutines, TRIMI and WAKEC1, listed in the rectangles represent the major tasks involved in trimming the rotor. Subroutine WAKEC1 calculates the influence coefficients for rotor-1. Subroutine TRIMI calculates the trim solution for the current wake stage.

Figure 2.25 expands subroutine TRIMI to further illustrate its function. In this figure, the outer rectangle represents a call to the subroutine TRIMI from subroutine TRIM. In TRIMI, the subroutine RAMF iterates on the rotor blade circulation and motion with fixed controls until the circulation and motion root-mean-square (rms) change from one revolution to the next is less than an input tolerance criterion. The function of subroutine TRIMI is to call RAMF which first calculates the forces and moments on the rotorcraft with the initial guess for the trim controls, then calculate a "derivative matrix" for use by the Newton-Raphson solution procedure, and finally uses a Newton-Raphson method to iteratively solve for the required rotor controls for the trimmed flight condition.

Figure 2.26 expands the RAMF subroutine to further illustrate its function. Subroutine RAMF first calculates the blade modes, then iterates on a circulation loop which in turn iterates on a motion loop. Inside the motion loop, an azimuth loop calculates the blade position and motion with subroutine MOTNB1. Then the blade aerodynamics are computed using AEROF1. Inside AEROF1, there is a radial loop that calculates the aerodynamics for all radial stations on the blade at the current azimuth.

If the low resolution indicial aerodynamics option is chosen, the effect of the option is to replace the azimuth loop in Figure 2.26 with the one in Figure

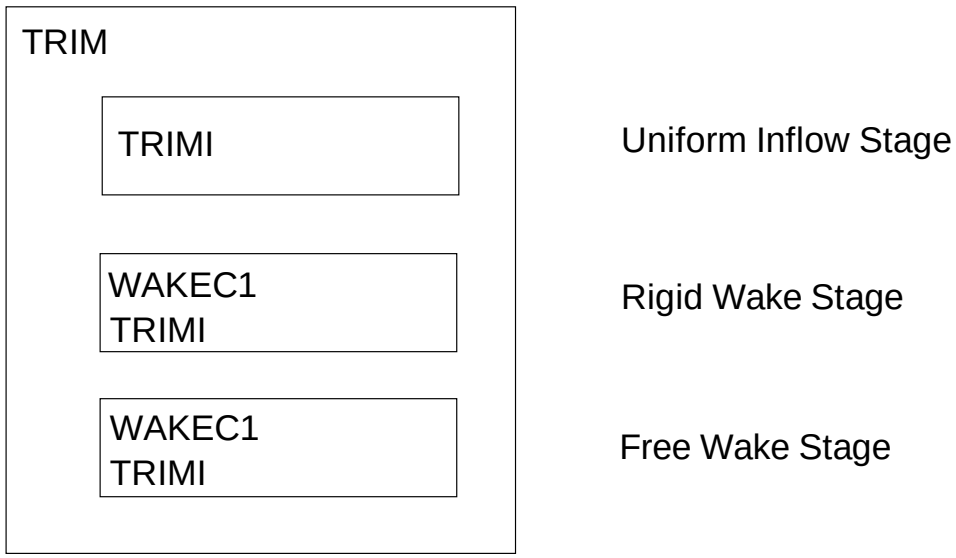


Figure 2.24: The basic trim loop of CAMRAD.Mod1.

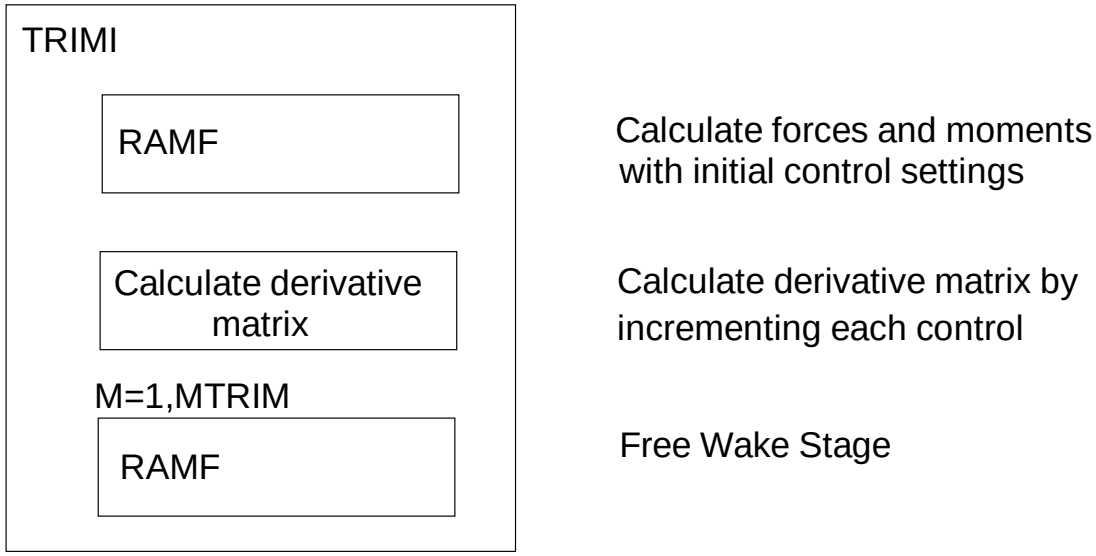


Figure 2.25: The TRIMI loop of CAMRAD.Mod1.

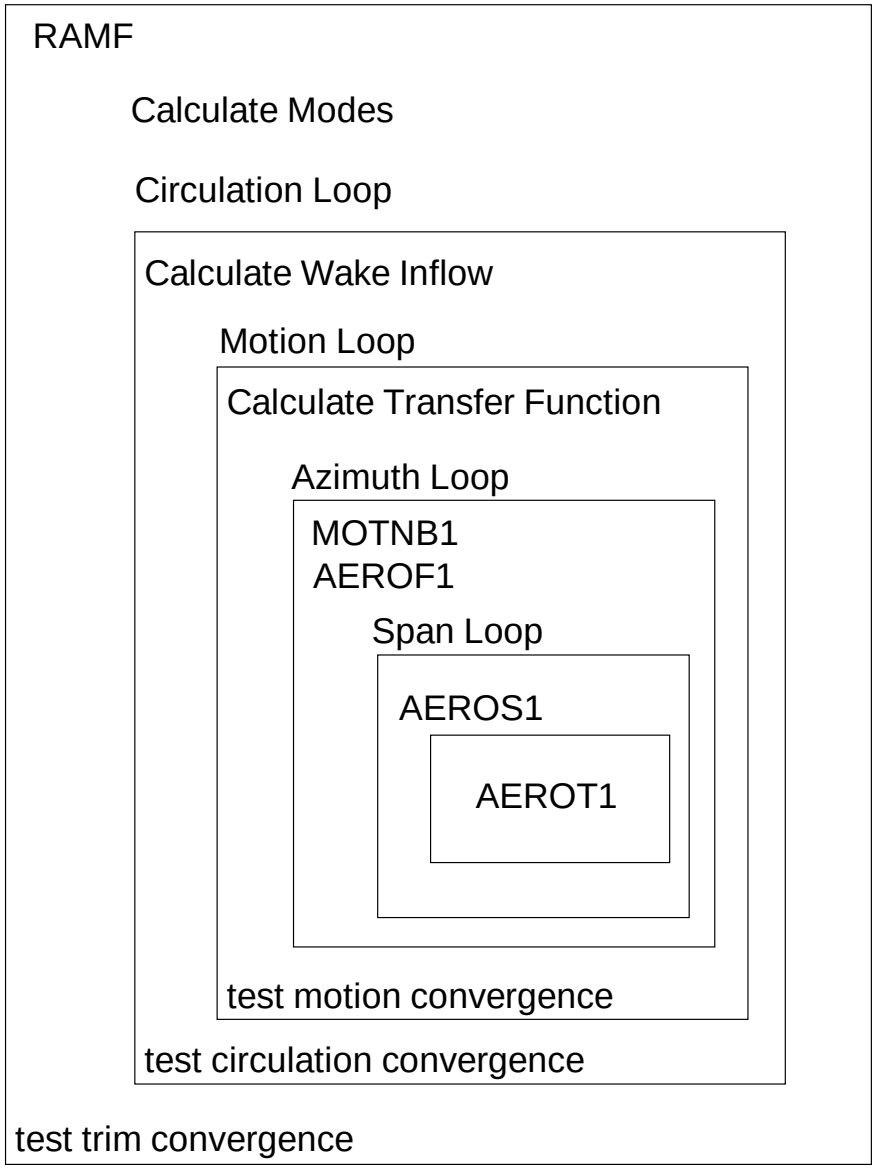


Figure 2.26: The RAMF loop of CAMRAD.Mod1.

2.27. This replacement loop has several differences compared to the original loop. First, a new subroutine, TWA1, calculates the near wake effects for the reference blade. Second, AEROF1 is replaced by AERBED1 to calculate the aerodynamics of the blade. Inside the radial loop in AERBED1, the subroutine AEROS1 has been replaced by AEROS1B, and a new subroutine SEPRATE1 has been added to calculate the effects of leading edge and trailing edge separation. More detail about the subroutines and the changes are given below.

2.16.2 Code Modifications

First, namelist reads for NLBED were placed in the subroutines INPTR1 and INPTR2 for rotor-1 and rotor-2, respectively. Contained in NLBED are the new parameters OPBED, HCOR, and ICURV. These and other parameters are listed in Chapter 5.

If OPBED1 or OPBED2 = 1, the parameter OPSTLL in CAMRAD.Mod1 is automatically set equal to unity internally, for the corresponding rotor (1 or 2), regardless of the input value for that parameter. This is done since the indicial aerodynamics includes a simple dynamic stall model. Therefore, for the indicial aerodynamics to use the 2-D airfoil tables correctly, the static stall option (OPSTLL = 1) must be chosen. This static stall option in CAMRAD.Mod1 equates to interpolating the 2-D airfoil table information at an unmodified (that is, unmodified by dynamic stall parameters) angle of attack. In addition, the subroutines INPTR1 and INPTR2 print a message to standard output reminding the user that the parameter OPSTLL has been set equal to unity internally.

Second, changes were made to the subroutines INPTW1 and INPTW2 so that if OPBED1 or OPBED2 = 1, the extent of the near wake (KNW) is checked for the corresponding rotor (1 or 2). If KNW is not equivalent to 90°, a warning message is printed to standard output. The value of KNW is not changed, and the program continues to execute. It is left to the user to correct the input parameter. In addition, INPTW1 and INPTW2 check that the original CAMRAD.Mod1 near wake is turned off (*i.e.,* WKMODL(2) thru WKMODL(5) are set = 0 so that the near wake is not "double counted"). If not, a warning message is printed to standard output. The parameters are not changed and the program continues to execute. It is left to the user to set the WKMODL input parameters correctly.

Third, if OPBED = 1, the TRIM subroutine calls the subroutine INITBED1 and INITBED2 for rotor-1 and rotor-2, respectively. These

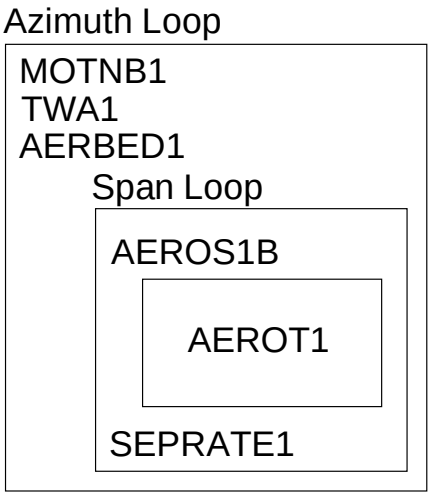


Figure 2.27: Replacement azimuth loop for the low resolution indicial aerodynamics option in CAMRAD.Mod1.

subroutines initialize the parameters needed in the indicial aerodynamics calculations.

Fourth, the calls to subroutine AEROF1 from MOTNR1 for rotor-1 and AEROF2 from MOTNR2 for rotor-2 were replaced by the following:

```
IF (OPBED1 .EQ. 1) THEN
IF (ITWA1 .EQ. 1) CALL TWA1(JPSI)
CALL AERBED1 (...)
ELSE
CALL AEROF1 (...)
ENDIF
```

and by:

```
IF (OPBED2 .EQ. 1) THEN
IF (ITWA2 .EQ. 1) CALL TWA2(JPSI)
CALL AERBED2 (...)
ELSE
CALL AEROF2 (...)
ENDIF
```

respectively.

The subroutines AERBED1 and AERBED2 are the indicial aerodynamics equivalent to the subroutines AEROF1 and AEROF2. The variables ITWA1 and ITWA2 are flags that are initially set to zero (internally) so that the indicial trailed near wake is not calculated for the case of uniform inflow. After the uniform inflow stage, the flags are set equal to unity (internally) so that the indicial trailed near wake will be included by calls to subroutines TWA1 for rotor-1 and TWA2 for rotor-2.

2.16.3 Subroutine Descriptions

1. INITBED1 and INITBED2

   Both of these subroutines initialize the parameters needed by the indicial aerodynamics subroutines (AERBED1 and AERBED2) at the beginning of the trim loop.

2. TWA1 and TWA2 (Trailed Wake Algorithm (TWA))

These subroutines use a modified version of the trailed wake algorithm of T.S. Beddoes (see Ref. [13]). These calculate the downwash at radial stations, RA, along the rotor blade at a specified azimuth station, JPSI, due to a trailed near wake system of vortices extending 90 ° behind the reference blade. The parameters WKMODL(2) thru WKMODL(5) need to be set equal to zero, as discussed earlier. The downwash due to the Kth vortex segment is determined at the Ith radial location. The "current vortex segments" are located at the radial segment endpoints, RAE. Placing the vortex segments at these locations is a modification to Beddoes' Trailed Wake Algorithm (TWA). The total instantaneous downwash, W_{new}, at a given radial station is then the sum of downwashes from all current vortex segments plus a contribution from the previous downwash, exponentially decreased by an amount equivalent to convecting the vortex location by one time step. The convection geometry has two options, straight or curved, as discussed earlier. The basic form of the equations (Ref. [14] and [15]) are as follows:

$$W_{new}(I,\psi) = \sum_{K}(X_w(I,K) + Y_w(I,K)) \tag{2.47}$$

$$X_w(I,K) = X_{w,old}(I,K)e^{-\phi\Delta t} + 1.359D_w \tag{2.48}$$

$$Y_w(I,K) = Y_{w,old}(I,K)e^{-4\phi\Delta t} - 0.359D_w \tag{2.49}$$

$$D_w = \left(\frac{\gamma(K)C_c}{4\pi h}\right)\frac{\frac{V\Delta t}{h}}{(1+\frac{V\Delta t}{h})^{1/2}} \tag{2.50}$$

$$C_c = \frac{h^2}{h^2 + {h_c}^2} \tag{2.51}$$

where h is the distance from Kth vortex to Ith radial station, h_c is the input HCOR core size, V is the freestream velocity encountered by section, $(= r + \mu\sin(\psi))$, Δt is the time step (1 azimuth step), $\gamma(K)$ is the strength of Kth vortex, ϕ is the decay factor depending on ICURV, h, and V.

Usage of the core factor C_c in Equations 2.50 and 2.51 is also a modification of the TWA of Beddoes. This factor is used to avoid the well

known singularity at the center of a potential vortex. This factor is a "Scully-type" viscous core factor. Once the instantaneous downwash, W_{new}, is known, a velocity deficit function is applied to allow the downwash to "build up", since the vortex does not instantaneously produce a downwash field. The resulting downwash, W_{app}, is then used by subroutine AERBED1. TWA2 does the same for rotor-2. The deficit function is of the form:

$$W_{app}(I,\psi) = W_{new}(I,\psi) - DEF(I) \tag{2.52}$$

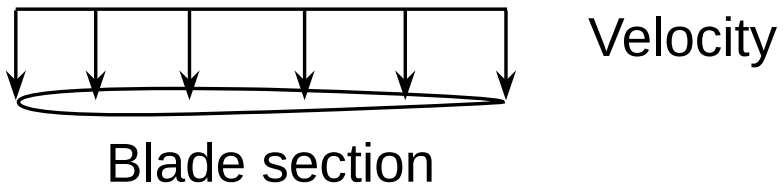
$$DEF(I) = DEF_{old}(I)e^{\frac{-2V\Delta t}{c}} + (W_{new}(I,\psi) - W_{new}(I,\psi_{old})) \tag{2.53}$$

where c is the local blade chord.

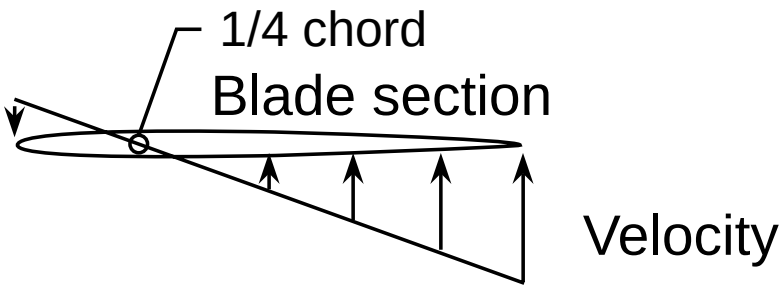
3. AERBED1 and AERBED2

Once the instantaneous downwash, W_{app}, is known from TWA1 and/or TWA2, all the velocities on the blade are known. The aerodynamic loads may then be calculated by AERBED1 and AERBED2 for rotor-1 and rotor-2, respectively. The indicial modifications basically involve replacing the angle of attack, with an effective angle of attack that includes the features of modified classical unsteady thin airfoil theory. That is, the angle of attack will be a quasi-steady angle of attack, minus a lift deficiency function in the form of an angle of attack deficiency function that accounts for the shed wake of an airfoil (see Ref. [14]).

The integrated downwash approach used to calculate the quasi-steady and of attack in Ref. [14], cannot be used correctly in the low resolution calculations of CAMRAD.Mod1. This integration is replaced by an angle of attack calculation that assumes the the downwash is comprised of a uniform downwash along the chord due to rotorcraft motion, wake inflow, and airfoil motion (except pitch rate) plus a linear downwash due to pitch rate motion of the airfoil. A picture illustrating this assumption is shown in Figure 2.28. The quasi-steady angle of attack needed in this method is the angle of attack based on the values of the velocity normal the the chord, U_n, the velocity in the chordwise direction, U_c, the velocity due to pitch rate, $U_{\dot{\theta}}$, and the velocity due to the near wake, W_{app}, at the 3/4 chord point:



(a) uniform downwash along chord due to rotorcraft motion, wake inflow and airfoil motion.



(b) linear downwash due to pitch rate motion of the airfoil.

Figure 2.28: Downwash components on the airfoil.

$$\alpha_{3/4} = \tan^{-1}\left(\frac{U_n + W_{app} + U_{\dot{\theta}}}{U_c}\right) \tag{2.54}$$

Once $\alpha_{3/4}$ is known, the deficiency functions found in Ref. [14, 15, 16] may be applied to calculate the effective angle of attack. Again, since the computation is performed in low resolution, the non-circulatory or impulsive terms defined in the mentioned References are not included. These impulsive components account for the apparent mass terms seen in classical unsteady aerodynamics and are high resolution effects. Thus including these in the low resolution sections would not be consistent with the assumption of low resolution analysis.

After computing the angle of attack, AERBED1 and AERBED2 call the subroutines, AEROS1B and AEROS2B, respectively. These sub-

routines, once called, calculate the lift, drag, and moment coefficients (C_l, C_d, C_m) by calling the subroutines AEROT1 and AEROT2, respectively, which interpolate the aerodynamic table information. These C_l, C_d, and C_m values have no time delay effects due to trailing edge or leading edge separation. These effects are accounted for in AERBED1 and AERBED2 which call the subroutines SEPRATE1 and SEPRATE2.

4. SEPRATE1 and SEPRATE2

   These subroutines calculate the lift and drag coefficients including the effects of trailing edge separation (TES) and leading edge separation (LES). The values from AERBED1 and AERBED2 before calls to SEPRATE1 and SEPRATE2 assume no TES or LES. The effect of TES is to lag the lift behind the angle of attack. This lag occurs because the chordwise location of the TES point does not instantaneously follow the changes in surface pressure during motions of the airfoil. This effect is explained in detail in the mentioned References. The TES theory coded in SEPRATE1 and SEPRATE2 follow these references. However, in the original publications, the TES chordwise point is assumed to be known *a priori* ; this is not the case here. Before the lagged TES point can be found, the unlagged TES point must be determined by solving the TES point equation in the references for the variable FNP. The resulting equation is:

$$FNP = \left[2 \left(\left| \frac{C_{l,c81}}{C_{l,potential}} \right| \right)^{1/2} - 1 \right]^2 \tag{2.55}$$

$$C_{l,potential} = \frac{2\pi\alpha}{(1 - M^2)^{1/2}} \tag{2.56}$$

   FNP is the value of the unlagged TES factor. $C_{l,potential}$ is the potential normal force coefficient. $C_{l,c81}$ is the normal force coefficient from the airfoil table lookup performed in AERBED1 (or AERBED2). FNP is then lagged and applied to calculate the TES contribution, C_{nnf}, to the normal force, as described in the mentioned References.

   SEPRATE1 and SEPRATE2 then calculate the LES contribution, C_{nnv}, to the normal force coefficient. This calculation is a simplified

version of the dynamic stall model developed in the mentioned References. This simplified version accounts for one vortex release during a dynamic stall event. This is a limitation only for deep, sustained stall.

The value of the normal force and chordwise force are then determined by the equations in the mentioned References. Since SEPRATE1 and SEPRATE2 need to return C_l and C_d, the chordwise and normal force coefficients are converted to these quantities before returning. It is assumed that the moment coefficient is appropriately represented by the value obtained from the interpolated table information from subroutines AEROT1 and AEROT2.

2.16.4 New Common Blocks

The following subroutines contain one or more new subroutines for the indicial aerodynamics: INITBED1, INITBED2, TRIM, MOTNR1, MOTNR2, AERBED1, AERBED2, TWA1, TWA2, INPTR1, INPTR2, SEPRATE1, SEPRATE2, INPTW1, and INPTW2. The new common blocks are as follows: BED1, BED2, BEDOLD1, BEDOLD2, BEDVEL1, BEDVEL2, BEDTWA1, BEDTWA2, BEDCON1, BEDCON2, BEDUSE1, BEDUSE2, BEDSEP1, BEDSEP2, DUCK1, and DUCK2. The list of variables in these new common blocks is extensive. For a listing of these variables, refer to the subroutines in the CAMRAD.Mod1 source code.

2.16.5 Known Caveats

1. The indicial aerodynamics model should not be used with the CFD interface since the TWA can not truncate the near wake as is required by the CFD interface.

2. Recently it has been discovered that there is a flaw in the TWA model in CAMRAD.Mod1. The symptoms are that the loads do not decrease as they should near the blade tip, and the loading is not smoothed appreciably in a "near wake" fashion and can cause problems in the large core calculations of the rollup model (discussed later). The exact cause of the problem is not known at this time. Therefore use of the low resolution indicial aerodynamics option is not recommended at this time.

2.16.6 Extensions to High Resolution

The indicial aerodynamics, for high resolution is implemented as a separate post-processor program to account for the near wake given a far wake "forcing function". The indicial post-processor and its implementation will be discussed in a later chapter.

2.17 Modifications for a Vortex Rollup Model

2.17.1 Introduction

In the original version of CAMRAD, the vortex wake consisted of several components. First, a near wake lattice model was used for the wake immediately behind the reference blade. The far wake for all blades (starting at the end of the near wake lattice model for the reference blade) consisted of a tip vortex and an inboard far wake "panel", usually represented as one shed and one trailed vortex in the center of the panel. Normally, both were present with large vortex core radii. The geometry of the inboard wake was determined by a prescribed/rigid wake model. The tip vortex distorted geometry was computed from either a prescribed/rigid wake model or from a free wake model. In calculating the influence coefficients for the tip vortex, it was assumed that the tip vortex geometry was determined solely by the blade tip position and the tip vortex geometry model. The tip vortex rigid wake geometry model assumes that the distorted wake geometry depends only on the convection of the vortex endpoints by the freestream velocity. The free wake model assumes that the wake geometry is determined by the freestream convection of the vortex endpoints and by the vortex self-induced velocities. Also, the tip vortex strength was determined as a function of the maximum bound circulation on the blade, regardless of the distribution of bound circulation on the blade. In order to incorporate more physical principals into the wake structure/geometry analysis, a new vortex model was introduced – the "rollup model". The rollup model modifies the wake geometry calculations which are used in determining the influence coefficients. The rollup model does not directly affect the free wake calculations, but it modifies the results.

Several parts of the rollup model will be discussed in the following sections. But first, the method of implementation will be discussed. Then the rolled-up vortex positions will be discussed, followed by a discussion of the vortex position "phase-in" models. Following that will be a discussion of

the multi-core vortex model used for both tip and secondary vortices. And finally, the vortex "spin" model will be discussed.

2.17.2 Method of Implementation

The rollup model (Ref. [3]) was introduced as an additional loop in the Trim loop of CAMRAD.Mod1. Originally, the Trim loop consisted of several wake stages: uniform inflow, rigid wake, and free wake. The rollup procedure was added as an additional step; the trim sequence now proceeds as follows: uniform inflow, rigid wake, free wake, free wake with rollup model (see Figure 2.29). At the end of the original three wake stages, the "free wake with rollup model" stage is entered. The first step in this stage is to calculate the rolled-up vortex positions for the tip and secondary vortices (this is part of the "ROLLUP Calcs"). These locations will be used to position the tip and secondary vortices relative to the original wake location determined by the free wake geometry calculations. Figure 2.30 shows the original positions of the wake endpoints and those that are shifted inward (in the rotor tip path plane) by an amount determined by the rollup calculations. There is also a vertical (perpendicular to the tip path plane) shift that will be discussed later. The second step is to calculate the strengths for each vortex core in the multi-core vortex model. This step is also part of the "ROLLUP Calcs." in Figure 2.29. With the vortex multi-core strengths and vortex locations known, the wake geometry used in the computation of the influence coefficients is then modified by shifting the wake endpoints by a calculated amount based on the rollup model and by using the multi-core model for the vortex core structure (labeled "Rollup Stage" in Figure 2.29). If needed, the first several wake endpoints may be phased-in to their final location over a prescribed wake age interval. Once the wake influence coefficients are known, the controls-motion-circulation interations proceed as before. Next, new airloads are calculated in the trim loop using the new rollup vortex locations, and the process is repeated for a user specified number of rollup-trim iterations.

2.17.3 Rolled-up Vortex Positions, Part 1

The rolled-up vortex model for the inward shift of the vortices ("Part 1") incorporates elements of the Betz inviscid rollup model (Ref. [17]) which was originally derived for trailed vorticity of fixed wing aircraft. Also included are adaptations from the works of Donaldson and Bilanin (Ref. [18]) and

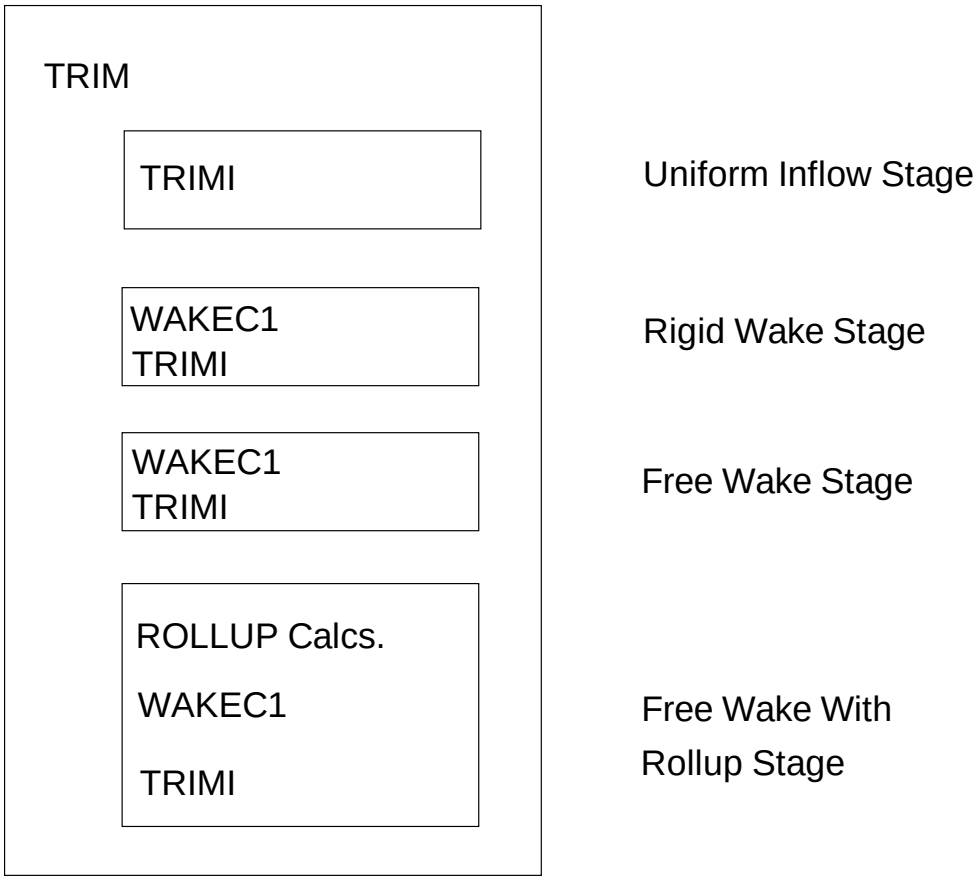


Figure 2.29: The TRIM loop with the new Rollup Calculations included.

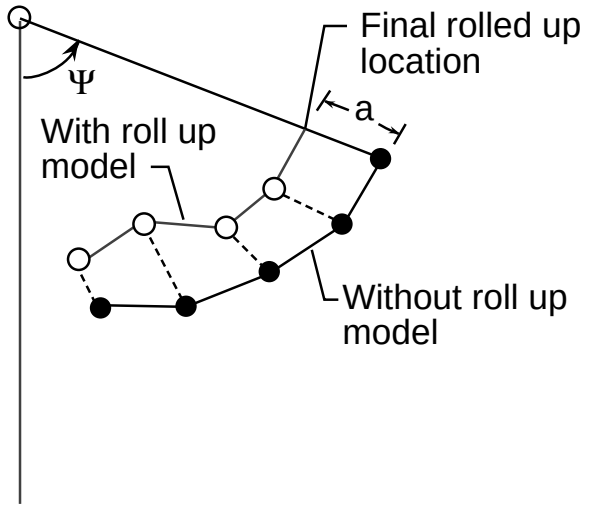


Figure 2.30: Wake-vortex intersection.

Bliss (Ref. [19]). In this model, the bound circulation distribution along the blade span is used to define and locate an axisymmetric tip vortex that has a circulation (strength) distribution which varies in the vortex radial direction (outward from the center of the vortex). The "rolled-up" position of the vortex (*i.e.,* the final location of the vortex) is calculated in the new model to be the fully rolled-up vortex position. This is similar to the Betz method in that the final spanwise location of the vortex far downstream is at the wing's spanwise centroid of vorticity. However, the rollup model also defines a "secondary" vortex, inboard of the tip vortex, when certain criteria imply that there should be more than one vortex trailed into the wake. The present analysis is limited to a maximum of two vortices (a tip and a secondary). The inboard wake treatment is the same as original CAMRAD. The model addresses only the final rolled-up location of the vortex endpoint in planes nominally parallel to the rotor disk. The final vertical rolled-up location will be discussed later. From the Betz rollup model, it is seen that the tip vortex far downstream of a fixed wing should be placed at the spanwise centroid of vorticity; whereas, in CAMRAD.Mod1, the tip vortex location, excluding the rollup effects, is determined by a free wake analysis. Due to the complication of the particular free wake geometry analysis in CAMRAD.Mod1, a direct modification of the free wake geometry analysis was not attempted. Instead, a superposition approach was taken.

In this approach, the free wake geometry is determined without the rollup model, then the geometry is adjusted to account for the rollup effects. This adjustment takes the form of an inward radial shift in the wake endpoint locations according to the rollup location calculations.

In the actual determination of the vortex rolled-up spanwise location, it is assumed that the circulation distribution is smooth radially. This leads to a smooth vorticity distribution. However, in typical helicopter BVI conditions, there can be significant spanwise circulation variations caused by perpendicular BVIs. It is assumed in this modeling that these spanwise variations caused by perpendicular BVIs do not contribute to sustained trailed vortices. This is, these spanwise loading variations contribute to local trailed vorticity, but do not create "long-lived" vortex filaments such as seen in a tip vortex. Since the modeling presented here is limited to two vortices (a tip and a secondary vortex), it is desirable to use a smooth bound circulation distribution to determine the final rolled-up positions of the tip and secondary vortices. Determining a smooth bound circulation distribution (*i.e.,* eliminating effects of the perpendicular BVIs) is the role of the "large core calculation". This large core calculation involves a "side" calculation whereby the wake influence coefficients are found using a "large" vortex core radius (typically about 0.3 rotor radii). Once the large core wake influence coefficients are known, the resultant loads and thus circulation may be computed. The circulation from this "side" calculation is dubbed the "large core circulation" (also known as the "fat core circulation" in [3]). Figure 9 from reference [3] illustrates a comparison between a full spanwise circulation distribution and the resultant large core spanwise circulation distribution (called the "fat core distribution" in Figure 9 of reference [3]). Here it is seen that the large core circulation has essentially removed the effects of perpendicular BVIs.

In addition to the large core calculation, the effects of blade rotation on vortex stretching must be taken into account. Figure 2.31 shows a rotating-blade tip vortex with straight line segments being emitted at a radial location $\overline{y}_t$. To illustrate a concept, vortex line segments, signifying vorticity shed from locations y, are shown outboard and inboard of $\overline{y}_t$. Each segment length depends on the spanwise location for a given azimuth step size and on the rotational and inflow velocity at location y. Each filament segment has an induced field velocity given by

$$\delta v = \frac{\delta, \delta l}{r} \tag{2.57}$$

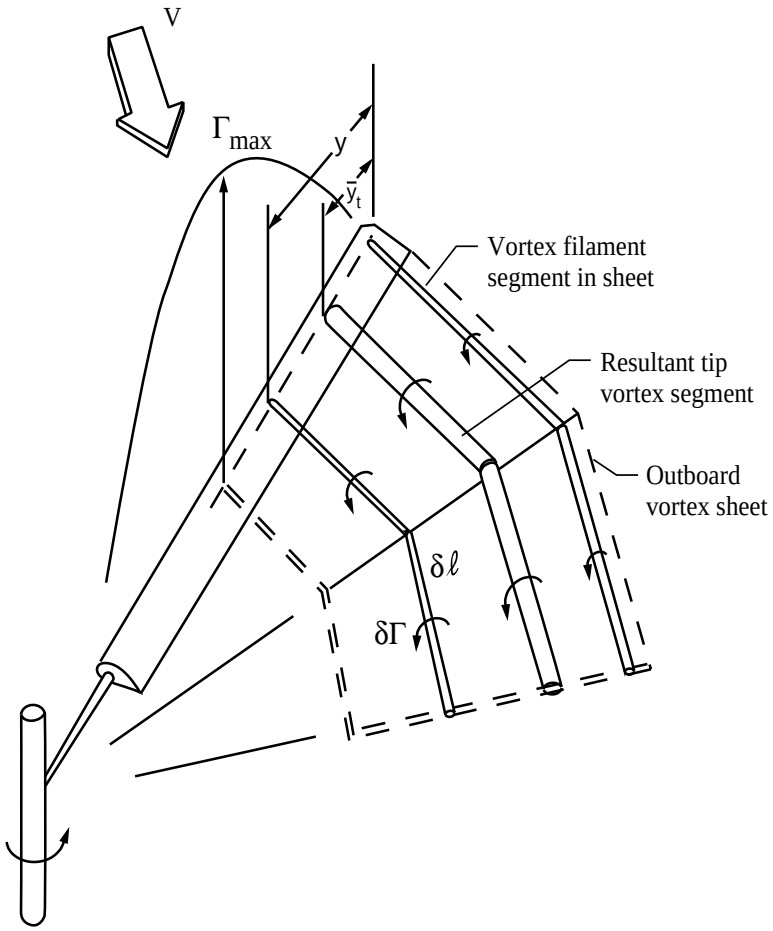


Figure 2.31: Rotor blade with line segment modeling of trailed vorticity due to bound circulation distribution.

where r is the perpendicular distance to the vortex. In the rollup concept considered here, the vorticity filaments are drawn to and are entrained into the vortex at location $\overline{y}_t$, with the vortex length being defined at $\overline{y}_t$. When this is done, stretching (or compression) of the vorticity must occur in proportion to the ratio of its original segment length to its new segment length at $\overline{y}_t$. This serves, in principle to maintain the same δv contribution at an observer located at a distance r in the wake. This concept is implemented by multiplying the large core spanwise circulation distribution by a weighting factor, F. The weighting factor for the large core spanwise circulation distribution for the tip vortex is as follows:

$$F = (1 - y) + \mu \sin \psi \tag{2.58}$$

The weighted large (fat) core spanwise circulation distribution is also shown in Figure 9 of reference [3]. This weighted distribution is subsequently used to determine the locations $\overline{y}_t$ and $\overline{y}_s$, the spanwise final rolled-up locations of the tip vortex and secondary vortex, respectively. The weighted large core spanwise circulation distribution, typically a smooth distribution, is used to locate the final rolled-up locations of the tip and secondary vortices. A spanwise vorticity centroid function can be found by the following equation:

$$\overline{y}(y) = \frac{1}{\gamma(y) - \gamma(0)} \int_0^y \frac{\partial \gamma(\eta)}{\partial \eta} \eta \partial \eta \tag{2.59}$$

$$\overline{y}(y) = \frac{1}{\gamma(y)} \int_0^y \frac{\partial \gamma(\eta)}{\partial \eta} \eta \partial \eta \tag{2.60}$$

where y is the blade spanwise coordinate pointing inboard starting at the tip, $\overline{y}(y)$ is the spanwise vorticity centroid as a function of y, and $\gamma(y)$ is the weighted large core circulation as a function of y. In practice, the derivative of γ is determined by a forward difference scheme starting at the tip of the blade, and the integral is calculated using the trapezoidal rule.

The current rollup model implements the above integral in three typical cases. "Case 1" (see Figure 2.32) assumes that the circulation distribution increases monotonically from a value of zero at the tip to a value of , $_{+max}$ at a radial station, y_{+max} (also labeled as point "A"). In this case, the minimum value of vorticity between the tip and y_{+max} is actually at y_{+max}, and the value of the vorticity there is zero. For this case, only a tip vortex is needed. In actual implementation and for coding convenience, the secondary vortex exists along with the tip vortex for all cases. However, in Case 1, the secondary vortex location is identical to the tip vortex location and its

strength is zero. The fully rolled-up position of the tip vortex is defined as the centroid of vorticity between the tip and point A from the integral below:

$$\overline{y}_t = \frac{1}{\gamma(A)} \int_0^{y=A} \frac{\partial \gamma(\eta)}{\partial \eta} \eta \partial \eta \tag{2.61}$$

where $\overline{y}_t$ is the vorticity centroid located between the tip and the point A, and γ is the weighted large core circulation, as discussed above.

If the function $\overline{y}(y)$ does not increase monotonically, it is assumed that two vortices are required; "Case 2" and "Case 3" deal with the two such possibilities. In Case 2 (see Figure 2.33), a non-zero minimum occurs in the vorticity distribution between the tip and y_{+max}. For this case, y_{+max} is labeled "B" and the location "A" is determined in the actual discretized radial solution as the radial station just prior to the station at which the function $\overline{y}(y)$ fails to increase monotonically. The final tip vortex rolled-up location here is determined, from Equation 2.61, to be the centroid of vorticity between the tip and point A. The secondary vortex location is found to be the centroid of vorticity between point A and point B as follows:

$$\overline{y}_s = \frac{1}{\gamma(B) - \gamma(A)} \int_{y=A}^{y=B} \frac{\partial \gamma(\eta)}{\partial \eta} \eta \partial \eta \tag{2.62}$$

If the value of $\overline{y}_s$ is ill-defined by this equation, the value of $\overline{y}_s$ is taken as a point half-way between points A and B.

For "Case 3" (see Figure 2.34), there exists a negative minimum in the circulation distribution, , $_{-max}$, between the tip and the positive maximum bound circulation, , $_{+max}$, which leads to an inflection point in the vorticity distribution. The maximum in circulation occurs at a radial station y_{+max}, labeled "B". And as before, point "A" (in the actual radially discretized solution) is located at the radial station just prior to the station at which the function $\overline{y}(y)$ fails to increase monotonically. Again, the tip vortex final rolled-up location is determined by the equation for $\overline{y}_t$ above between the blade tip and point "A"; the secondary vortex final rolled-up location is determined by the equation above for $\overline{y}_s$ between the points "A" and "B". In this case, the tip and secondary vortices will have the opposite circulation sense (*i.e.,* the tip vortex will have a negative strength and the secondary will have a positive strength).

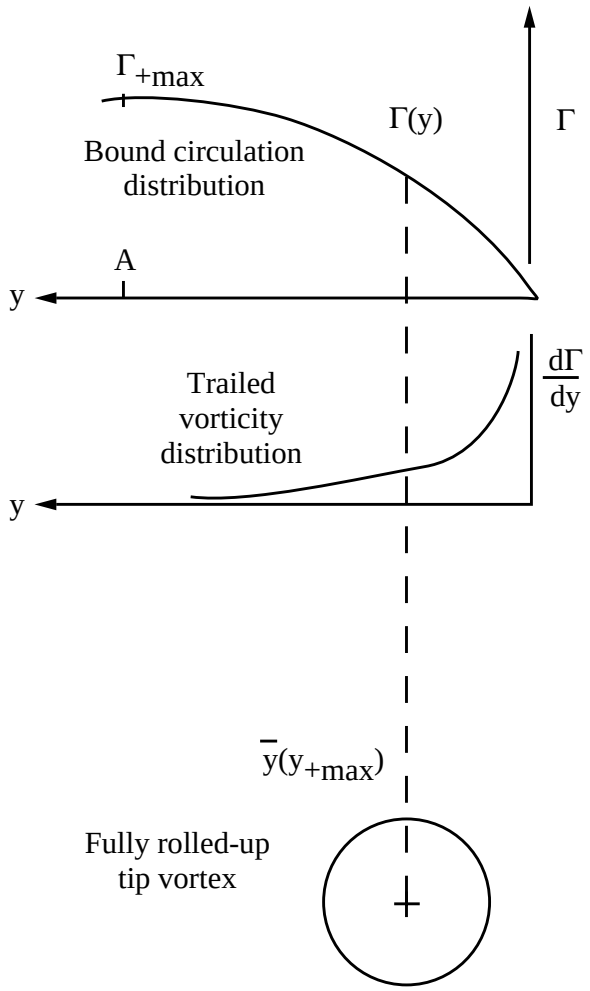


Figure 2.32: Final rolled-up location - Case 1.

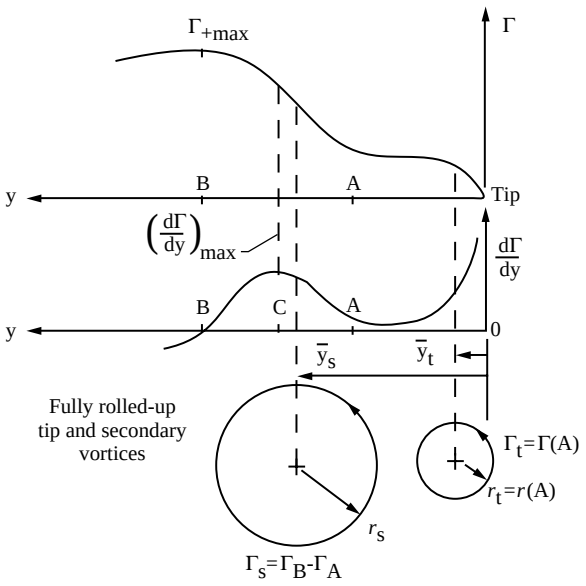


Figure 2.33: Final rolled-up location - Case 2.

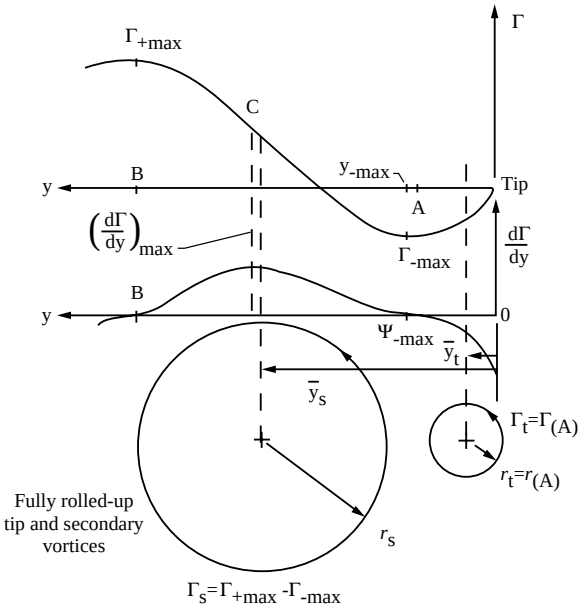


Figure 2.34: Final rolled-up location - Case 3.

2.17.4 Rolled-up Vortex Positions, Part 2

The previous subsection discussed the final rolled-up location of the tip and secondary vortices in a plane nominally perpendicular to the rotor shaft. The final vertical rolled-up location in now considered. Previously in CAMRAD.Mod1, the tip vortex was assumed to originate at the tip of the blade. The original location of the tip vortex was defined by locating the blade tip in space, then setting the coordinates of the zeroth wake age to that coordinate. At subsequent wake ages, the wake endpoint location in space was determined by a combination of the blade tip position when the endpoint was "deposited" in the wake, the convection of the endpoint by the free stream, displacement of the endpoint by a free wake distortion vector, and displacement of the wake endpoint by an amount prescribed by the rollup model. The previous subsection discussed the additional endpoint displacement term from the rollup in the horizontal plane. This subsection discusses the additional term for the vertical coordinate. For the rollup model, this vertical term is needed to account for the fact that certain vortices do not leave from the tip of the blade.

For a vortex (tip or secondary) that is inboard of the blade tip, the final vertical rolled-up position term is equal to the vertical location at the spanwise final rolled-up location on the blade. For example, of there were no coning or bending of the blade, the vortex final rolled-up location would be constructed from a spanwise location, and a vertical final rolled-up location equal to zero. This is because the tip and the spanwise rolled-up location are at the same vertical coordinate for this scenario. As another example, if a rigid, articulated blade were coned upward, the spanwise rolled-up location would be below the tip of the blade. Therefore, the wake endpoint would be shifted downward by the difference in these two coordinates (see Figure 2.35). With this additional term, the vortex appears to eminate from the blade at the location marked with an "X" in the figure, instead of the location marked with an "O".

2.17.5 Rolled-up Vortex Position Phase-in

In the previous subsections, the final rolled-up locations of the tip and secondary vortices are discussed. Provisions are made to allow the vortex to migrate to or "phase-in" to the final rolled-up position from some initial spanwise location on the blade. Several models for this have also been developed and implemented in CAMRAD.Mod1. First, a simple functional

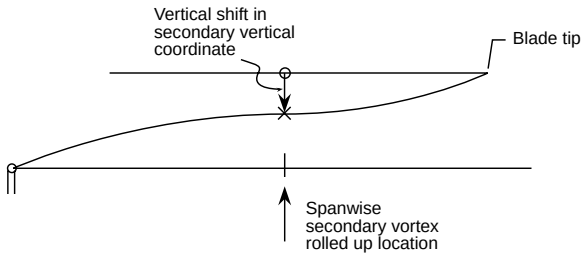


Figure 2.35: Z-coordinate for rolled-up vortices in rollup model.

form was implemented to specify a phase-in scheme. Second, a model based on fixed-wing work of Spreiter and Sachs was attempted. This second model was abandoned after initial testing. In CAMRAD.Mod1, the coding for this model still exists, but the variable controlling its usage has been hard coded such that the option cannot be used. Thus, the Spreiter/Sachs option will not be discussed in this documentation.

The prescribed phase-in model in CAMRAD.Mod1 has several forms which can be applied to the tip and/or the secondary vortices. First, the variable ISECPH in namelist NLROLL determines if the secondary vortex will be involved in the phase-in process. Normally, the secondary vortex does not participate in the phase-in process as it is assumed that the secondary vortex will "form" and remain at the location $\overline{y}_s$. Therefore, the secondary vortex is normally considered to always be at its final rolled-up position. The tip vortex, on the other hand, is assumed to be "created" at some location on the blade outboard of the secondary vortex. The tip vortex is assumed to then phase-in to (*i.e.,* migrate to) its final rolled-up position. Figure 2.36 illustrates a wing with a "tip" vortex trailed. In the figure, note that at some downstream wake age, the vortex is located at its final rolled-up position. Conceptually, the phase-in function is smooth and continuous. In practice, the phase-in function is applied at given wake segment endpoints of fixed wake age; each vortex segment between these endpoints is a straight line. Also, note that the vortex originates from a user-specified fraction of the final rolled-up location relative to the tip of the blade. Between these two locations, a tenth-order polynomial of wake age, with user-specified coefficients, is used to phase-in the tip vortex. Normally, the initial location is assumed to be at the tip of the blade and the phase-in occurs linearly in age (by using only the first term in the phase-in polynomial). In a typical case, the phase-in is assumed to be complete (*i.e.,* the tip vortex is at its

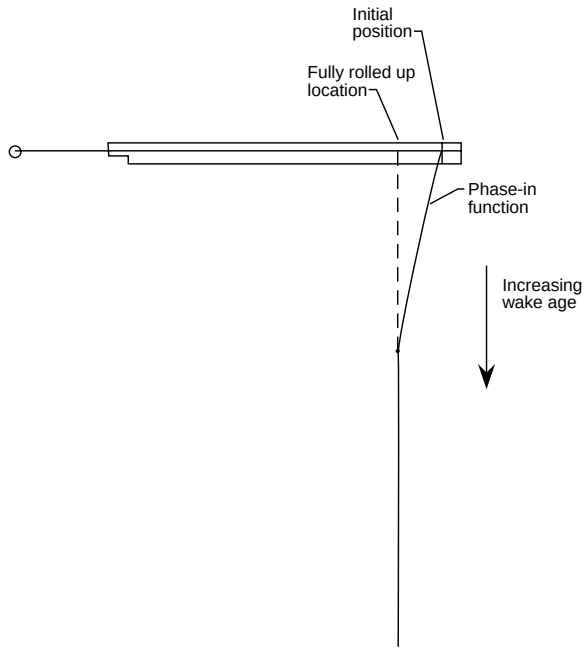


Figure 2.36: Rolled-up location "phase-in" model.

final rolled-up location) after one rotor revolution.

2.17.6 Multi-Core Vortex Model (Tip Vortex)

Another feature of the rollup model is the "multi-core vortex model". From classical fluid mechanics, the velocity field induced by an infinite ideal vortex (a solution to Laplace's equation) is as follows:

$$v = \frac{,}{2\pi r} \tag{2.63}$$

where , is the circulation associated with the vortex, and r is the radial perpendicular distance to the point where the velocity is desired. This vortex is irrotational (except at the center), and has a velocity distribution that approaches infinity as the center of the vortex is approached. In "real" fluids, viscous and turbulent effects become dominant near the center of the vortex and thus prevent infinite velocities at the vortex center. Historically, many models have been introduced to account for the viscous effects near the center of the vortex. These models typically assume a particular velocity

distribution based on a vortex core size, r_c. For example, a Rankine vortex model for the velocity is similar to that of an ideal vortex for $r \geq r_c$, but for $r < r_c$, a solid body rotation is assumed. This assumption leads to a linear velocity increase from zero at the center of the vortex to the value $v = \frac{\Gamma}{2\pi r_c}$ at the edge of the vortex core. Another example is a model dubbed here as the "Scully" vortex model and has an induced velocity distribution as follows:

$$v = \frac{,}{2\pi r}\left(\frac{r^2}{r^2 + {r_c}^2}\right) \tag{2.64}$$

where , , r, and r_c are defined as discussed above. The term in the parantheses is known as the core factor and it serves to smoothly transition the induced velocity profile v from the ideal vortex to zero at the center of the vortex, with a peak induced velocity at the distance r_c from the center of the vortex. The functional form of the core factor is arbitrary to a certain degree, but does provide a smooth transition in the velocity profile as a function of distance from the vortex center, unlike the Rankine vortex that has an abrupt change from a linear function to a $1/r$ function at the edge of the vortex core (that is, at $r = r_c$).

Previously in CAMRAD.Mod1, the tip vortex core was modeled (typically) using the Scully vortex model above. The tip vortex core size was a single, constant input quantity. The present vortex core modeling for the tip and secondary vortices implements a multi-core model which ties the strength of the tip and secondary vortices to the large core circulation distribution on the blade at each azimuth. Figure 2.37 illustrates the model of the structure of the far wake vortices which depend on the large core circulation distribution at each blade azimuth station. The vortices shown are taken as fully developed; the intermediate rolling-up process and aging process is not modeled here. The structure of the tip and secondary vortices are represented in Figure 2.37 as "sets" of concentric vortices of varying radius.

In reality, the tip vortex strength would vary in a continuous manner radially outward from the center of the vortex. However, a discretized solution is desired in CAMRAD.Mod1. Thus, the strength of the tip vortex will be calculated at a set number of user input core sizes. Up to ten input core sizes ($(r_c)_p$) are possible (p is the index of the input core sizes; $p = 1, 2, 3, ..., 10$). The only requirement on these core sizes is that they are spaced in such a manner that they adequately resolve the viscous core region.

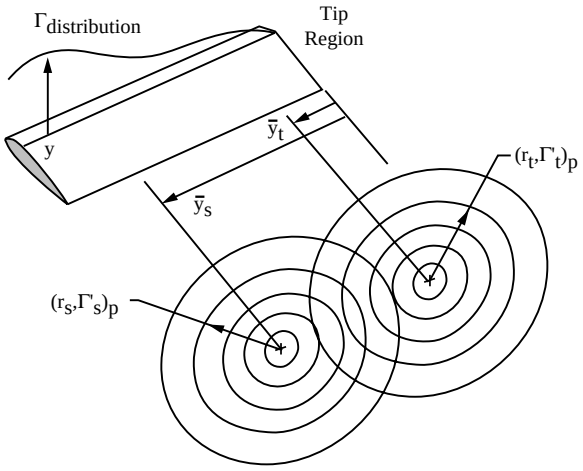


Figure 2.37: Multi-core model (tip and secondary) shown at fully rolled-up locations.

Using the discretized version of the spanwise vorticity centroid function as discussed earlier in this section, the spanwise vorticity centroid function becomes, $\overline{y}(y_i)$, where y_i are the discretized radial locations. The discretized, weighted large core spanwise circulation, $\gamma(y_i)$, is assumed to be related to $\overline{y}(y_i)$ as shown in Figure 2.38. The circulations are linearly interpolated to the actual user input core sizes, $(r_c)_p$, where the subscript p is the index of the user input core sizes. This assumed relation holds for the tip vortex until the point labeled "A" is reached for any of the three possible "Cases" discussed previously. Any input core size that is outside of this interpolation range is assigned a strength of zero. The interpolated circulation values are represented by the symbol γ_p, where, again, p is the index of the user input core sizes. The strength, γ'_p for the vortex of core radius r_p is determined as the difference in circulation values between vortices of core size r_p and r_{p-1}. For the tip vortex, this difference is as follows:

$$(\gamma'_t)_p = (\gamma_t)_p - (\gamma_t)_{p-1} \tag{2.65}$$

Note that if $p = 1$ the term $(\gamma_t)_{p-1} = 0$.

The most inner core radius $(r_t)_p$ for the tip ($(r_s)_p$ for the secondary) represents an approximate viscous core radius. This inner core size may be determined in two ways. First, it may be a constant, user specified core size that would be the smallest core size expected from the particular rotor being

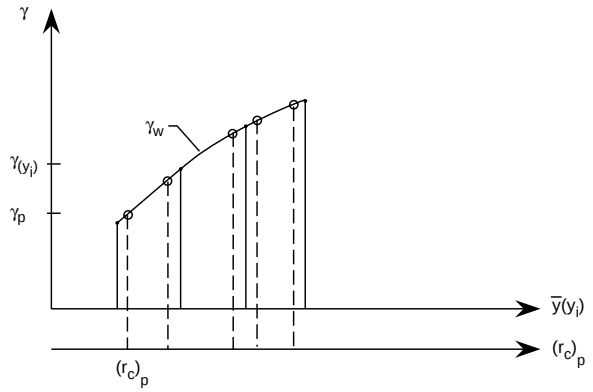


Figure 2.38: Large core, weighted circulation as a function of $\overline{y}(y_i)$ and input core sizes $(r_c)_p$.

examined. Or second, it may be determined by an empirical model. (The empirical model is discussed later as an option to the multi-core model.) The above definition of the multi-core model concept allows one to match nearly arbitrary large core circulation distributions to trailed vortex strength distributions. It is seen that although some choices can be made in the code to define the inner core radii of the tip and secondary vortices (as options), the importance of the core radii as tuning parameters is significantly reduced when compared to the single core model.

In order to maintain the same δv contribution at an observer at r in the wake, the strengths of γ'_p are "unweighted" by the following factor:

$$F = \frac{1}{(1 - \overline{y}_t) + \mu \sin \psi} \tag{2.66}$$

The strengths of the all of the vortices in the multi-core representation of the tip vortex are now known, and the velocity field of this vortex can be calculated. The following formula is used to calculate the velocity field at any point due to influence of the multi-core tip vortex:

$$v = \sum_{p=1}^{p=P} \frac{(\gamma'_s)_p}{2\pi r} \left(\frac{r^2}{\sqrt[n]{(r^{2n} + (r_c)_p^{2n})}} \right) \tag{2.67}$$

where p is the summation index, P is the number of vortex cores in the multi-core model, r is the perpendicular radial distance to the vortex center, $(r_c)_p$ is the p^{th} user input core size, n is a user input integer used to vary the

vortex core model and the term in parantheses is the core factor. Note the similarity in the functional form of the velocity profile to the form of the Scully type vortex model discussed previously. Also, note that for different values of n many of the common core models are recovered. For example, if $n = 1$, the Scully model is retained. If $n \to \infty$, the Rankine vortex model is recovered. Typically a value of $n = 2$ is used.

2.17.7 Multi-Core Vortex Model (Secondary Vortex)

As discussed earlier, if a "Case 2" or "Case 3" situation is detected, a secondary vortex is assumed to exist at a final rolled-up spanwise location $\overline{y}_s$. Like the tip vortex, the strengths of the multiple concentric vortex cores that compose the secondary vortex are needed to calculate the velocity field due to the presence of the secondary vortex. Whereas the tip vortex multi-core strength distribution was related to the function $\overline{y}(y)$, the secondary multi-core distribution is related directly to differences in spanwise circulation on the blade. First, a spanwise origin, labeled point "C" in Figures 2.33 and 2.34, is needed that lies between points A and B. The amount of vorticity outboard of point C and the amount of vorticity inboard of point C are assumed to be additive in the multi-core model for the secondary vortex. That is, the vorticity outboard of point C will provide a vortex strength distribution in the multi-core model of the secondary, as will the vorticity inboard of point C. These effects are the superimposed. Point C is located by finding the maximum rate of change of the weighted large core circulation, $|d\gamma/dy|$. In practice, the derivative is calculated using a forward difference scheme:

$$\frac{d\gamma}{dy} \approx \frac{\gamma_i - \gamma_{i-1}}{y_i - y_{i-1}} \tag{2.68}$$

where the subscript i starts at the location just inboard of point A and continues until point B. Also, in practice, the location of the maximum vorticity is determined by successively testing the calculated value of $|d\gamma/dy|$, as above, with 1.1 times the previously calculated value. If the radial station associated with the maximum slope is found in this manner, it is saved as point C. If a maximum is ill-defined by this process, point C is determined by the radial station midway between the radial stations assigned to point A and to point B. Figure 2.39 illustrates points A, B, and C for a generic "Case 3". The vertical axis is the relative circulation value at the given radial stations y_i. The second (lower) horizontal axis represents the interpolation

of the relative circulation values onto distances equivalent to the user input secondary vortex core sizes $(r_c)_p$ outboard (between A and C) and inboard (between B and C) of the point labeled C. First, the circulation at each of these core sizes $(r_c)_p$ in the outboard region is determined by the relative circulation distribution at that radial station. In the case shown (*i.e.,* Case 3), these values will be negative since the circulation outboard of point C is less than that at point C. Next, the circulation inboard of point C is determined in a manner similar to that used in the outboard region. These circulations at equivalent vortex core radii $(r_c)_p$ are then subtracted from one another as follows:

$$\gamma_p = \gamma(r_c)_{p,inboard} - \gamma(r_c)_{p,outboard} \tag{2.69}$$

where now, γ_p is the circulation value associated with the input user core size $(r_c)_p$. The strength (vorticity) of each of these core sizes is now taken as the difference in the circulation between the cores as follows:

$$(\gamma_s')_p = (\gamma_s)_p - (\gamma_s)_{p-1} \tag{2.70}$$

where p is, as before, the user input vortex core size index. As before, if $p = 1$, the term $\gamma_{p-1} = 0$. Again, in order to "unweight" the strengths of the multi-core model, the $(\gamma_s')_p$ terms are multiplied by the following function F:

$$F = \frac{1}{(1 - \overline{y}_s) + \mu \sin \psi} \tag{2.71}$$

Note that except for the term $\overline{y}_s$, this F is just the inverse of the F used to weight the large core spanwise circulation distribution. Now the induced velocity field due to the secondary multi-core vortex is calculated in the same manner as the velocity field induced by the tip multi-core vortex:

$$v = \sum_{p=1}^{p=P} \frac{(\gamma_s')_p}{2\pi r} \left(\frac{r^2}{\sqrt[n]{(r^{2n} + (r_c)_p^{2n})}} \right) \tag{2.72}$$

2.17.8 Multi-Core Vortex Model Options

There are actually three multi-core model options in CAMRAD.Mod1. The first, described previously, is the default option in CAMRAD.Mod1 when using the rollup model. This default option uses an array of constant core sizes for the multi-core tip vortex and the multi-core secondary vortex.

The second option, controlled by the variable ICORYCB in namelist NLROLL, is to use a single, empirically determined core size for the multi-core model. This option is referred to as the "single variable core, multi-core model". This option uses the following empirically derived core sizes:

$$(r_c)_t = 0.015 + 0.075\overline{y}_t \tag{2.73}$$

$$(r_c)_s = 0.015 + 0.075(\overline{y}_s - y_A) \tag{2.74}$$

where $(r_c)_t$ is the inner most tip vortex core size ($p = 1$), $(r_c)_s$ is the inner most secondary vortex core size ($p = 1$), $\overline{y}_t$ is the fully rolled-up location of the tip vortex as calculated above, $\overline{y}_s$ is the fully rolled-up location of the secondary vortex as calculated above, y_A is the radial location of point A, y is the distance from the tip of the blade to the inboard point y on the blade. With these core size definitions, the core sizes for the tip and secondary vortices vary azimuthally. The strengths of the core for each the tip vortex and secondary vortex are determined as before with the index $p = 1$.

The third option, refered to as the "variable multi-core, multi-core model", is a hybrid of the two previous models. First, the user defines a set of constant radius vortex core sizes as done in the default option. Then the "variable multi-core, multi-core model" will first calculate the distribution of strength in the constant core sizes as in the default option. With the strength distribution known, the empirical core sizes discussed previously are assumed to be the minimum viscous core radius. With this assumption, the "closest" core size in the constant core size array is reset to the minimum core size. Any strength inside of the minimum core size is summed and assigned to the newly defined minimum core size. Since the minimum core size will be between two core sizes in the constant core size array, an interpolated amount of strength is also removed from the core size that is just larger that the minimum core size. The amount that is added to the minimum core size is then removed from the larger core. The purpose of this last step is facilitate a smooth transition between two core sizes as the minimum core size increases and decreases azimuthally. For example, if the minimum core size is only slightly smaller that a particular core size in the constant core size array, the minimum core will receive all strength inside its radius and most of the strength from the core size to which it is closest. Subsequently, the core size that is closest to the minimum core size will have almost zero strength. This is illustrated in Figure 2.40. The user input constant core sizes in this Figure are $r1, r2$,and,$r3$. The empirically determined core size is $(r_c)_t$. In practice, the core size $r2$ is set to the value $(r_c)_t$. This

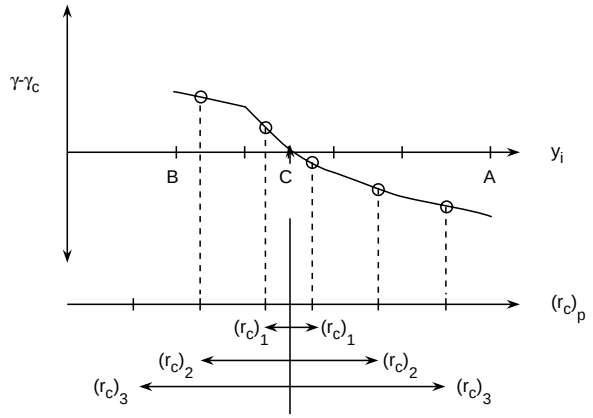


Figure 2.39: Spanwise locations of A, B, and C in multi-core model for secondary vortex.

core $(r_c)_t$ will have the summed strength of (1) $r1$ and (2) $r2$ and (3) an interpolated strength between cores $r2$ and $r3$. The strengths of cores $r1$ and $r2$ are subsequently set to zero for the remainder of the calculations; the strength of core $r3$ is reduced by the same amount that was gained by $(r_c)_t$ in the interpolation step above (step (3)).

2.17.9 Multi-Core Model Caveats

The wake influence coefficient calculations in the original version of CAMRAD were developed for a constant vortex core size. In the multi-core model options where a core size varies azimuthally, these same influence coefficient calculation precedures are still used. That is, the effect of a linearly varying vortex core size on the value of influence coefficient for each wake segment endpoint is not taken into accounted.

2.17.10 Vortex Pair Spin Model

The present use of the Scully free wake model in CAMRAD.Mod1 puts constraints on the way that the wake geometry can be modified to account for effects of the rollup model. At present, any particular wake endpoint of a tip and secondary vortex is associated with a wake endpoint from the free wake geometry calculation. This free wake geometry endpoint is adjusted after the free wake geomtry calculation to include the rollup model vortex

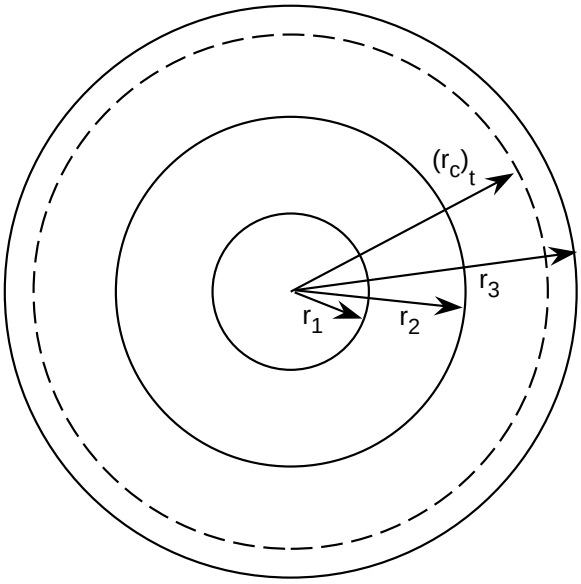


Figure 2.40: Variable multi-core, multi-core model interpolation schematic.

position calculations (discussed previously) in the influence coefficients calculations. Note that the free wake geometry calculation does not account for the fact that in some parts of the rotor wake, there exist both a tip and secondary vortex. Figure 2.41 depicts a portion of a wake filament which splits into a tip and secondary vortex over part of its length. The two vortices in this case are shown with opposite (but not necessarily equal) strengths, , $_s$ and , $_t$. A model was implemented to account for the mutual influence of the tip and secondary vortices on each other. This model, called the "spin" model, is used to rotate the wake endpoints to a new position based on a two-dimensional model of the influence of the two isolated vortices on each other. For example, in Figure 2.41, both vortices will tend to rotate upward due to the others' influence. If the secondary vortex were stronger than the tip vortex, the tip vortex would be moved higher that the secondary. Figure 2.42 shows a two dimensional scenario used to calculate effects of the spin model. The following geometric relations can be derived using the figure:

$$\vec{P}_t = \vec{P}_t(r_t, z_t) \tag{2.75}$$

$$\vec{P}_s = \vec{P}_s(r_s, z_s) \tag{2.76}$$

$$\theta_z = \tan^{-1}\left(\frac{z_t - z_s}{r_t - r_s}\right) \tag{2.77}$$

$$\zeta_t = \frac{r_t}{\cos\theta_z} \tag{2.78}$$

$$\zeta_s = \frac{r_s}{\cos\theta_z} \tag{2.79}$$

$$\Delta\zeta = |\zeta_t - \zeta_s| \tag{2.80}$$

where $\vec{P}_t$ and $\vec{P}_s$ are the vectors to the tip and secondary vortex endpoints at a given azimuth location. At a given azimuth location, the velocity influence of each vortex endpoint on the other is calculated using a two dimensional infinite multi-core vortex model:

$$V_t = \sum_{p=1}^{p=P} \frac{(\gamma_s)_p \Delta\zeta}{(\Delta\zeta)^2 + ((r_c)_p)_{secondary}{}^2} \tag{2.81}$$

$$V_s = \sum_{p=1}^{p=P} \frac{(\gamma_t)_p \Delta\zeta}{(\Delta\zeta)^2 + ((r_c)_p)_{tip}{}^2} \tag{2.82}$$

where V_t is the velocity influence at the tip vortex due to the secondary multi-core vortex, and where V_s is the velocity influence at the secondary vortex due to the tip multi-core vortex. This model does not include the effect of the exponent n that was discussed in a previous section (*i.e.,* a "Scully" type vortex core model is still assumed here). Using the above quantities, new quantities can be calculated:

$$\zeta_o = \frac{V_s\zeta_t - V_t\zeta_s}{V_s - V_t} \tag{2.83}$$

$$\omega_o = \frac{V_t\Omega}{\zeta_t - \zeta_o} \tag{2.84}$$

$$\cos\psi = \frac{y_t - y_s}{\left|\vec{P}_t - \vec{P}_s\right|} \tag{2.85}$$

$$\sin\psi = \frac{x_t - x_s}{\left|\vec{P}_t - \vec{P}_s\right|} \tag{2.86}$$

$$\theta_o = \frac{z_t - z_s}{\left|\vec{P}_t - \vec{P}_s\right|} \tag{2.87}$$

$$z_o = (\zeta_o - \zeta_t)\sin\theta_o \tag{2.88}$$

$$r_o = (\zeta_o - \zeta_t)\cos\theta_o \tag{2.89}$$

where Ω is the rotor rotational rate, x, y, z with subscripts t, s are the coordinates of the wake endpoints being examined for the tip and secondary vortices, respectively. It is necessary to identify three "Cases" in order to correctly apply the spin model to a particular situation. (Note that these cases are not associated with the three Cases discussed in earlier subsections.) Once the spin Case is identified, several parameters can be evaluated. If first statement is true, then the following two quantities are set:

$$Case1: \quad \zeta_o < \zeta_s < \zeta_t \tag{2.90}$$

$$\theta_{t,o} = 0. \tag{2.91}$$

$$\theta_t = \omega\tau_c\tau \tag{2.92}$$

$$\theta_{s,o} = 0. \tag{2.93}$$

$$\theta_s = \omega\tau_c\tau \tag{2.94}$$

$$Case2: \quad \zeta_s < \zeta_t < \zeta_o \tag{2.95}$$

$$\theta_{t,o} = \pi \tag{2.96}$$

$$\theta_t = \omega\tau_c\tau + \pi \tag{2.97}$$

$$\theta_{s,o} = \pi \tag{2.98}$$

$$\theta_s = \omega\tau_c\tau + \pi \tag{2.99}$$

$$Case3: \quad \zeta_s < \zeta_o < \zeta_t \tag{2.100}$$

$$\theta_{t,o} = 0. \tag{2.101}$$

$$\theta_t = \omega\tau_c\tau \tag{2.102}$$

$$\theta_{s,o} = \pi \tag{2.103}$$

$$\theta_t = \omega\tau_c\tau + \pi \tag{2.104}$$

where τ_c is a user defined multiplier for the rate of spin. Another user input constant, τ_o defines the wake age at which the spin calculations begin. For example, of $\tau_o = 0.0$, the spin calculations begin immediately. Once the spin Case has been identified, the following relations are applied to modify the tip and secondary vortex endpoint locations to account for the spin "effects" discussed above:

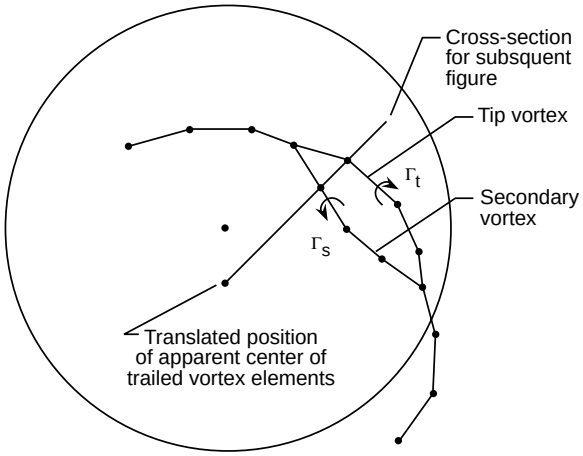


Figure 2.41: Vortex "spin" model schematic.

$$\vec{P}_t' = \vec{P}_t + |\zeta_t - \zeta_o| \begin{pmatrix} \cos(\theta_o + \theta_t) - \cos(\theta_o + \theta_{t,o}) \cos\psi \\ \cos(\theta_o + \theta_t) - \cos(\theta_o + \theta_{t,o}) \sin\psi \\ \sin(\theta_o + \theta_t) - \sin(\theta_o + \theta_{t,o}) \end{pmatrix} \tag{2.105}$$

$$\vec{P}_s' = \vec{P}_s + |\zeta_s - \zeta_o| \begin{pmatrix} \cos(\theta_o + \theta_s) - \cos(\theta_o + \theta_{s,o}) \cos\psi \\ \cos(\theta_o + \theta_s) - \cos(\theta_o + \theta_{s,o}) \sin\psi \\ \sin(\theta_o + \theta_s) - \sin(\theta_o + \theta_{s,o}) \end{pmatrix} \tag{2.106}$$

where $\vec{P}_t'$ and $\vec{P}_s'$ are the new coordinates of the vortex endpoints included in the spin model, $\vec{P}_t$ and $\vec{P}_s$ are the new coordinates of the vortex endpoints before the spin model.

2.18 Namelist Reading Subroutine Changes

2.18.1 Introduction

Many new variables and capabilities have been added to CAMRAD.Mod1. The original version of CAMRAD included a BLOCKDATA capability for input of many code parameters that were not normally altered in the course of a prediction task. Variables that are changed routinely were modified using namelist inputs. The newly added variables in

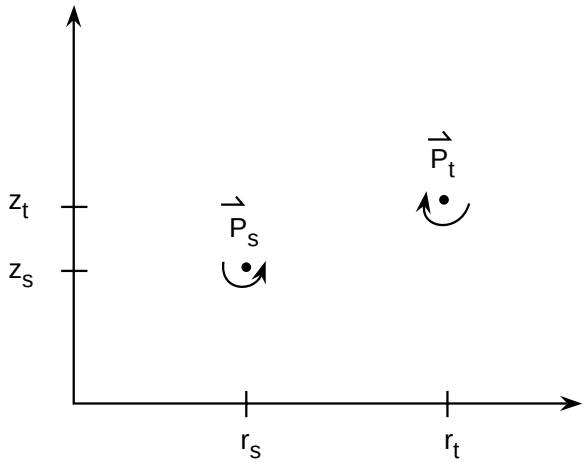


Figure 2.42: Vortex locations for "spin" model from previous figure.

CAMRAD.Mod1 are normally given values using the namelist inputs using new namelists. In some cases, existing namelist reading subroutines were modified to input the new namelists; in other cases, new namelist reading subroutines were added. This section describes each of the changes and additions. Some of the changes are discussed in other sections, but this section is intended to provided a concise listing of the changes.

2.18.2 Subroutine INPTN

This subroutine controls the reading of all namelists and reads namelist NLTRIM. Two new variables, FACTM and OPMXFWG, were added to NLTRIM; default values for these two variables were also added. Two new namelist reading subroutine calls were added to this subroutine. The new subroutine INPTCFD was added immediately following the call to INPTW1. INPTCFD reads the namelist NLCFD which contains parameters controlling the CFD interface. Immediately following the call to subroutine INPTCFD, a new call to subroutine INPTM1 was added. INPTM1 reads the namelist NLMEAS which controls the usage of measured blade motion and measured C_n information in CAMRAD.Mod1.

2.18.3 Subroutines INPTR1 and INPTR2

INPTR1 is one of the most highly modified of the namelist reading subroutines. Originally this subroutine read only namelist NLRTR. In addition, it now reads the following namelists (1) NLHHC , (2) NLHHC2, (3) NLHIRES, (4) NLBED, and (5) NLSWP. NLHHC and NLHHC2 contol the usage of Higher Harmonic Control; NLHIRES controls the usage of the HIRES section of CAMRAD.Mod1; NLBED controls the usage of the Indicial Aerodynamics in the low resolution section of CAMRAD.Mod1; and NLSWP controls the usage of the aerodynamic sweep correction in the high and low resolution sections of CAMRAD.Mod1. All of these namelists must exist in the script file after NLRTR, even if they are empty. Many of the variables in these namelists are given default values in INPTR1. All of the above changes apply to subroutine INPTR2.

2.18.4 Subroutine INPTW1 and INPTW2

INPTW1 originally read only the wake namelist NLWAKE. Now, it also reads the namelists NLBURST and NLROLL. NLBURST controls the parameters in the vortex bursting model. NLROLL controls the usage of the vortex rollup model. INPTW1 now also performs many checks to be sure that certain incompatible variable combinations are not attempted by the user. Some non-functioning options are also flagged in this subroutine with error messages printed to the standard output file. All of the above changes apply to subroutine INPTW2.

2.19 Fuselage Aerodynamic Tables

2.19.1 Introduction

CAMRAD.Mod1 was modified to include changes to the fuselage aerodynamics made by Sikorsky under contract to NASA Langley. These changes, originally developed by Sikorsky and subsequently implemented in CAMRAD.Mod1, modify the fuselage aerodynamics in CAMRAD.Mod1 by replacing the empirical analysis by "look-up" tables. With this model, the empirical aerodynamic formulae for the wing, the body, the horizontal tail, and the vertical tail, are replaced by a table "look-up" of aerodynamic characteristics for a new "wing-body-tail". These aerodynamic characteristics for the wing-body-tail are applied at a wing-body-tail location instead of at separate locations for the wing, the body, the horizontal tail, and the

vertical tail. See reference [20] for details. Several changes, mainly common block size corrections, were made to the original Sikorsky implementation.

2.19.2 Code Changes

1. The new subroutine BODYAT from Sikorsky was added.
2. The variable WBTTAB was added to the common block BADATA in subroutines BODYA, BODYAT, BODYF, FILEI, INITB, INPTN, PERF, PRNTB, ROTNET, and TRIMP.
3. The variable WBTTAB was initialized to zero in subroutine INPTB.
4. A subroutine call to BODYAT was added to subroutine BODYF if WBTTAB is set to one.

2.19.3 Extensions to High Resolution

This modification has no bearing on the high resolution calculations.

2.20 Machine Dependencies

2.20.1 Time

Original modifications to CAMRAD.Mod1 were made on a VAX with a VMS operating system. The code was ported to a DEC ALPHA workstation operating with a UNIX (OSF/1) operating system. Changes were made to the code to allow proper operation of the time and date stamps used in the code output. In the executive program, CAMRAD, in the input preparation program, BLOCKFILE, and in the airfoil preparation program, AIRFOIL, all calls to the system subroutine ITIME were changed to the following:

CALL ITIME (IHOUR, IMIN, ISEC)

and the information is encoded into a variable, such as JDTIME, using the following:

```
    INTEGER JDTIME(2)
    ENCODE (8,901,JDTIME) IHOUR, IMIN, ISEC
901 FORMAT (I2, 1H:, I2, 1H:, I2)
```

In the subroutine TIMER, all calls to ITIME were changed to the following:

```
INTEGER IARRAY(3)
CALL ITIME (IARRAY)
```

and the requested program time in seconds is calculated using:

```
T = IARRAY(3)*3600. + IARRAY(2)*60. + IARRAY(1)
```

2.20.2 Date

In the executive program, CAMRAD, the input preparation program, BLOCKFILE, and the airfoil preparation program, AIRFOIL, all calls to the system subroutine IDATE were changed to the following:

```
CALL IDATE (IMONTH, IDAY, IYEAR)
```

and the information is encoded into a variable, such as JDDATE, using the following:

```
    INTEGER JDDATE(2)
    ENCODE (8,902,JDDATE) IMONTH, IDAY, IYEAR
902 FORMAT (I2, 1H/, I2, 1H/, I2)
```

2.20.3 Dimension Statements

Some FORTRAN compilers do not allow a dimension statement in a subroutine to be similar to the following examples:

```
DIMENSION A(1)
DIMENSION AA(NM,1)
```

Therefore, in order to make the code more portable, the dimension statements of the form of the prior examples were changed to the following:

```
DIMENSION A(*)
DIMENSION AA(NM,NM)
```

when the intent of the dimension "1" is to pass an arbitrary length array to the subroutine. This replacement with the character "*" tells the program to pass the array with the same dimensions as it has in the calling routine. In the second example, the actual dimension, NM, is used instead of the "1".

2.20.4 Debug and Input Data prints

Originally the "INPUT DATA" section printed to standard output depended on the variable LEVEL. For example, if LEVEL = 0, the wake input quantities were printed for uniform inflow only. To get the input data printed for the free wake portion, one was required to execute the program through the free wake stage. This dependency was eliminated by removing several IF statements in the subroutines PRNTW1 and PRNTW2. (Actually, they were "commented out", not deleted.) Now, all the input parameters for the wake are printed, regardless of the value of LEVEL.

The COMPLEX variable KEPSI was removed from the debugging namelist DBINC in subroutine INITC. This was done because the DEC ALPHA did not allow a COMPLEX data type to be printed using a namelist with other data types present. This removal impacts only the printing of the variable KEPSI in the initialization stage of the program if the debugging flag DEBUG(3) = 2.

2.20.5 File Handling

The subroutine FILEV was altered to operate in a UNIX environment. The original OPEN statements that have VAX-specific keywords, such as READONLY, were altered to remove non-standard keywords. In addition, the original logical file names, determined from VMS system calls, are now obtained from the UNIX operating system by the system call "getenv (environmentvariable, actualfilename)". The environment variables are as follows: INPUTFILE, AFTABLE1, AFTABLE2, RESTARTFILE, EIGENFILE, LPOUTDB, LPOUTPP, LPOUTPUT, LPOUTLIN, NLINPUT, AFTABLE, AFDECK1, ..., AFDECK10.

Other files are opened without machine dependent elements in the OPEN statements. Filenames in these OPENs are hard coded, such as "ALPHAP.DAT". Other unit numbers are used in various places and each of these unit numbers must be linked to a file via the "ln -s" command. All unit numbers used are listed in the User's Manual.

2.20.6 Logicals and DATA statements

In subroutines that create printer-plots, DATA statements were originally used to set values of variables that are declared as a LOGICAL*1 type. This is not allowed by all compilers. To make the code more portable, these declarations were modified. INTEGER variables were created corresponding to

the affected LOGICAL variables. These INTEGER variables have the same name as the LOGICAL variables with an "II" prepended. Then the INTEGER variables are initialized using the DATA statement and the LOGICAL variables are EQUIVALENCEd to these INTEGER variables. For example the declarations of the type:

```
LOGICAL*1 THING
DATA THING /1H-/
```

were changed to:

```
INTEGER IITHING
DATA IITHING /1H-/
EQUIVALENCE (IITHING, THING)
```

This type of change was made in the following subroutines: BODEPP, FLUTM, GEOMP1, GEOMP2, HISTPP, NOISR1, NOISR2, POLRPP, STABM, TRCKPP, TRIMP, and in the airfoil preparation program.

2.21 Miscellaneous Changes and Bug Corrections

1. In the subroutines FILEV, an error was corrected. The variable BKLDAT was corrected to BLKDAT. This error effected only the reading of a BLOCKDATA program as an ASCII input file. Normally, the input data is read as a binary file.

2. In the subroutines FLUTR1 and FLUTR2, a bug was corrected. The variable OPTION was corrected to OPFLOW. This bug had an effect only in the flutter analysis.

3. Two subroutines, PRNTHR1 and PRNTHR2, were added to print information contained in namelists NLHHC, NLHHC2, NLHIRES, NLBED, NLSWP, NLBURST, NLROLL, and NLCFD to standard output if the parameter NPRNTT = 1 in namelist NLTRIM.

4. The common block WORK was defined originally in two different places in the code. The WORK common blocks in GEOMP1 and GEOMP2 were renamed WORKG1 and WORKG2 to eliminate the bug. These common blocks were not used outside of GEOMP1 and or GEOMP2.

5. Where practical, many of the exponent operators, ** , from the most inner loops in CAMRAD.Mod1 have been changed to multiplications since in some instances, multiplication is many times faster than exponentiation.

6. Two subroutines, INITHR1 and INITHR2, were added to subroutine INIT to initialize radial HIRES parameters. Addition of these subroutines reduces the number of input parameters required to run the high resolution part of CAMRAD.Mod1.

Chapter 3

HIRES

3.1 Introduction and Solution Procedure

3.1.1 Introduction

For many rotorcraft analysis tasks, a low temporal (azimuthal) and spatial (spanwise) resolution analysis of the aerodynamics and dynamics of the rotorcraft is sufficient. For example, in a performance analysis of an entire vehicle, 15 degree azimuthal resolution and 15 radial stations on the blade span may be sufficient to determine the average forces generated by the rotor. These average forces may be used to determine a steady state configuration for the rotorcraft. There are several comprehensive rotorcraft tools available to analyze these configurations and flight conditions. One such example of a comprehensive rotorcraft code is the original version of CAMRAD. Written using a low resolution azimuthal and spanwise discretization, limits were originally imposed such that azimuth step sizes of 15° and larger were required to be used (10° and larger if not using the free wake analysis). Also, the maximum number of spanwise locations on the reference blade was thirty. But, a high radial resolution and high azimuthal resolution are needed to calculate high resolution airloads for use in prediction of loading noise and Blade-Vortex Interaction (BVI) noise.

Experience has shown (Ref. [8]) that an azimuthal resolution of 1°, and a radial discretization of 75 radial stations along the span, are often adequate to resolve the BVI unsteady aerodynamic loading required for the acoustic analysis. One possible (conceptual) method to increase the resolution of a low resolution code is to simply redimension all relevant arrays in the comprehensive code and execute the code at that increased

resolution. One factor limiting the success of such an approach is the large increase in computation time (CPU time) that would be required to trim the rotor. Some of the increase in computation expense would be the fact that, internal to the code, many revolutions of the rotor(s) are required to converge the internal motion and circulation loops of the code. Increased resolution in these loops would slow the convergence of the comprehensive code. In addition, free vortex wake models can become unstable as the vortex length decreases, due to a higher resolution trim solution. However, even if this approach were practical, the general complexity of the free wake algorithm implemented in CAMRAD would make such global free wake code changes prohibitive.

Another approach is to allow the rotorcraft to achieve a trimmed state at a low resolution, effectively not changing the comprehensive code, then apply a post-trim analysis to "reconstruct" a higher resolution solution from the low resolution solution. A true post-processor type of analysis would apply a "stand-alone" code to the output information of the low resolution performance code. For example, a separate post-processor code might interpolate all output information up to a higher temporal and spacial resolution, or might execute a totally different analysis, such as a CFD analysis, to operate on the low resolution information. In this chapter, a modified post-processor type of analysis is used for the high resolution portion of the code. Since the high resolution portion of the code, as implemented, is not a completely stand-alone code, this approach may be thought of as a hybrid post-processor analysis.

With a hybrid post-processor analysis, there are a number of paths available to determine the high resolution loading solution. One such path, using an external CFD code, was discussed in the section on the CFD interface in Chapter 2. This current chapter will focus on another method that has become known as HIRES. This method includes all of the coding invoked at the end of the trim loop in CAMRAD.Mod1 (excluding the CFD interface, the ROTONET/WOPWOP interfaces, the Flutter analysis, and the Transient analysis).

3.1.2 Solution Procedure

As discussed previously, a calculation of high resolution wake induced velocities, and thus loading, would not be feasible in the trim loop of CAMRAD.Mod1. Thus, a modified post-processor type of analysis (known as HIRES) was added to the end of the trim loop to process the information

into a high resolution solution. This solution procedure effectively uses the low resolution wake solution and kinematically reconstructs the wake at azimuthal locations between known low resolution locations. HIRES also uses the modal analysis results from CAMRAD.Mod1 to define the blade position at a given high resolution azimuth location. With this information known, interpolations are use to calculate the high resolution information. The interpolation technique conceptually involves (1) linearly interpolating the wake geometry in wake age and azimuth, such that the tip vortices kinematically translate from one known low resolution location to the next known low resolution location, (2) linearly interpolating the blade shape radially between known low resolution collocation points on the blade, (3) linearly interpolating the blade bound circulation to a higher azimuthal and radial resolution. With these quantities known, the wake influence coefficients may be determined at a high azimuthal and radial resolution. Then, with the influence coefficients known, the wake induced velocities, and thus the loading, may be calculated at a high azimuthal and radial resolution. The actual details of the HIRES implementation can differ from the above conceptual implementation. These differences are due to practical issues such as code execution time constraints, coding convenience, *etc.* The details of the solution procedure are presented here.

One task involved in obtaining high resolution airloads is to calculate the wake induced velocities at a high resolution. This calculation requires knowledge of the blade position, the bound circulation, and the wake geometry at a high azimuthal (temporal) and radial (spatial) resolution. The azimuthal blade position may be determined at any azimuthal location without modifications to the code. This is true because CAMRAD.Mod1 uses a modal analysis and stores the harmonics of the generalized coordinates of the blade mode shapes at user specified low resolution (spanwise) collocation points on the reference blade. Storing these quantities in this manner allows one to compute the coordinates of any low resolution (spanwise) collocation point at any azimuth station. A high spanwise resolution is achieved by linearly interpolating coordinates amongst the low resolution radial collocation points. In order for this interpolation technique to closely approximate the true blade shape, it is required that as many low resolution spanwise collocations points be used as possible.

The high resolution bound circulation is obtained initially by linearly interpolating the low resolution bound circulation to a high azimuthal and radial resolution. Since the bound circulation is used in determining the wake induced velocity, the interpolated values will be used to initiate the

high resolution solution.

In determining the wake geometry at a high resolution, the locations of the tip (and possibly secondary) vortices and the locations of the inboard wake elements are needed. For the tip (and secondary) vortices, the high resolution vortex end points are located by kinematically translating from one low resolution location to another. Linear interpolation is used to locate these intermediate endpoints (see later section entitled "Vortex Segment Location"). In addition, the inboard wake elements are located by found by linear interpolation between low resolution endpoint locations. (see later section entitled "Vortex Segment Location"). With the high resolution blade coordinates known, the high resolution bound circulation known, and the high resolution wake geometry known, the wake induced velocities and aerodynamic loading may be computed at a high resolution.

3.1.3 Implementation of Solution Procedure

Historically, the first high resolution modification was the development of a far wake model. This model included only the effect of the tip vortex starting at the blade tip and existing for the same number of wake spirals as in the low resolution portion of the code. As such, the entire inboard vortex wake and lattice near wake were not modeled. This model was used in Reference [8]. At the next HIRES development stage, the inboard far wake elements were modeled. Several new models were also implemented in the far wake solution including a vortex segmentation model to effectively smooth the straight 10° vortex segments automatically as needed.

In the subsequent HIRES development stage, two methods were developed simultaneously to account for the near wake behind the reference blade. One method was implemented as an option inside the high resolution portion of CAMRAD.Mod1 and the other was developed as an independent post-processing code. The near wake model inside the high resolution portion of the code is a vortex lattice model roughly similar in nature to the vortex lattice model in the low resolution portion of the code. This method will be discussed in a later section of this chapter. The other method, which is an independent post-processor, called the Indicial Post-Processor (IPP), implements a time domain indicial aerodynamics formulation derived from the methods of T. S. Beddoes and Gordon Leishman (Ref. [13, 14, 15, 16]). This method will be discussed in Chapter 4.

As discussed earlier, Figure 1.1 shows an outline of the CAMRAD.Mod1 code with HIRES represented as a rectangle. The arrow into the HIRES

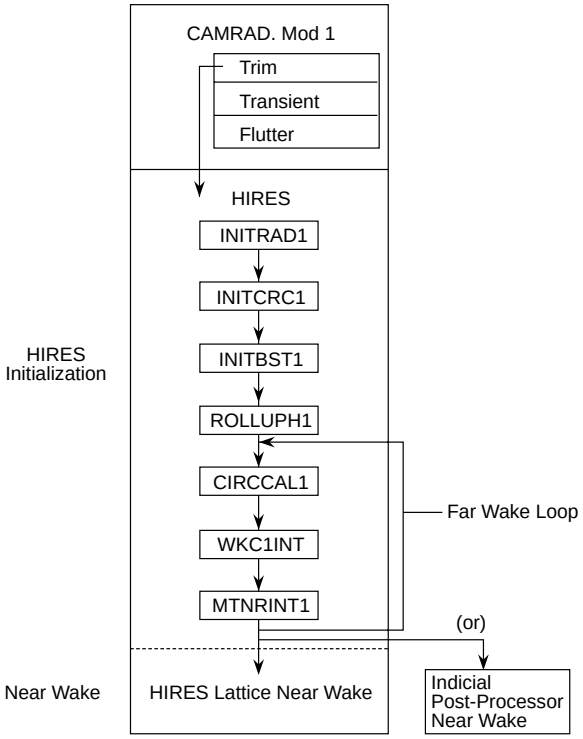


Figure 3.1: CAMRAD.Mod1/HIRES Schematic

rectangle conceptually shows that HIRES is executed after the Trim loop of CAMRAD.Mod1. Figure 3.1 expands the HIRES rectangle of Figure 1.1, to illustrate the procedures followed in HIRES. First, the rotor is trimmed using the low resolution portion of the code (CAMRAD.Mod1), then the low resolution circulation is linearly interpolated azimuthally and radially, to a high resolution. The wake influence coefficients are then determined using only the "far wake". The far wake consists of 10 ° agewise vortex segments and an inboard (rigid) far wake model (to be discussed later). With the far wake influence coefficients determined, the wake induced velocities and airloads (associated with the far wake only) may be calculated. If determined to be necessary (by the user), this "Far Wake Loop" (see Figure 3.1) may be repeated. Normally, only once through the Far Wake Loop is required. At the end of this iteration loop, the user has the option to use the high resolution vortex lattice near wake or the IPP as discussed earlier.

If the user chooses the vortex lattice near wake model (the option is

actually chosen at the beginning of the code execution by setting appropriate parameters in namelist NLHIRES), the execution of HIRES automatically will continue from the Far Wake Loop into the "HIRES Lattice Near Wake" box shown in Figure 3.1. Otherwise, the code will stop execution at the end of the Far Wake Loop. This is the path that is chosen (that is, stopping after the Far Wake Loop) if the IPP is to be used for the high resolution near wake calculations. Note that if a second rotor is also included, the entire high resolution procedure may be repeated for rotor-2. This is accomplished by replacing the "1" in each box of Figure 3.1 by a "2".

3.2 Initialization

There are a number of variables used in HIRES that are linearly interpolated from the low resolution solution of CAMRAD.Mod1 to a high resolution (azimuthally and/or radially) and do not change during the HIRES phase of CAMRAD.Mod1. These interpolated variables are calculated before the Far Wake Loop to avoid unnecessary interpolations during the computationally intense portions of HIRES. This calculation stage before the Far Wake Loop is called "Initialization". (Note that some initialization of HIRES parameters takes place at the same time the low resolution parameters are being initialized. These initialized parameters are ones that depend on the values of user input low resolution parameters such as blade chord, twist, *etc.* These high resolution parameters are found by radially interpolating (linearly) the low resolution parameters to the high resolution radial stations provided by the user in namelist NLHIRES (variable RAEINT). This initialization of the radial parameters is accomplished in subroutines INITHR1 and INITHR2 which are called just after the call to subroutine INITR2 in subroutine INIT.)

As seen in Figure 3.1, there are four subroutines before the far wake loop is begun. The four subroutines handle the initialization of the high resolution parameters (*e.g.,* bound circulation, vortex burst parameters, *etc.*) that depend on the low resolution solution. The functions of the initialization subroutines are described below.

Subroutines INITRAD1 and INITRAD2 (for rotor-1 and rotor-2, respectively) linearly interpolate, azimuthally and radially, the additional velocity distribution over the rotor disk from the tunnel/fuselage correction method to a high resolution for use in the high resolution induced velocity calculation.

Subroutines INITCRC1 and INITCRC2 (for rotor-1 and rotor-2, respectively) initialize the high resolution bound circulation and maximum bound circulation, azimuthally and radially, by linear interpolation. For the far wake calculations, only the maximum bound circulation is needed at a high resolution. The high resolution bound circulation distribution will only be used if the internal high resolution vortex lattice near wake model is used.

Subroutines INITBST1 and INITBST2 (for rotor-1 and rotor-2, respectively) initialize the high resolution tip vortex burst parameters from the low resolution solution.

Subroutines ROLLUPH1 and ROLLUPH2 (for rotor-1 and rotor-2, respectively) calculate parameters needed for the high resolution tip vortex rollup calculations. These parameters are linearly interpolated from the low resolution rollup calculations to high resolution values.

3.3 High Resolution Far Wake

3.3.1 Introduction

After the initialization of the HIRES parameters, the far wake loop is executed. A subroutine, CIRCCAL1 (or CIRCCAL2 for rotor-2), is called to lag the circulation in the far wake loop (if the far wake loop is to be executed several times, that is). This lag factor is used similar to all of the other iteration lag factors in CAMRAD.Mod1. However, experience has shown that normally one Far Wake Loop iteration is sufficient and the lag factor is simply set to unity.

After the subroutine CIRCCAL1 is called, the far wake influence coefficients are calculated at a high resolution by the subroutine WKC1INT (WKC2INT for rotor-2). This subroutine comprises the bulk of the far wake high resolution calculations. Once the influence coefficients are calculated, the wake induced velocities are calculated using the high resolution circulation distribution. These velocities are then combined with velocities due to blade rotation, blade motion, *etc.* at each spanwise section. Once these total far wake velocities are known, the angle of attack and Mach number may be computed. These angles of attack and Mach numbers are then used in an airfoil table to determine the airloads due to far wake effects.

3.3.2 Far Wake Influence Coefficients

The high resolution far wake influence coefficients are used to calculate the normalized induced velocity due to each vortex element in the wake system at a particular point on the rotor blade. For the high resolution far wake influence coefficients calculation, the far wake consists of the bound vortex of all blades except the reference blade, the inboard far wake of each blade, and a far wake tip vortex from each blade. The bound vortex for each blade, excluding the reference blade, consists of a straight line vortex from root to tip with a vortex core size of 25% of the mean blade chord. This model follows the low resolution model of the bound vortices. The inboard far wake consists of a two vortex segments, one in the trailed direction and one in the shed direction, on each vortex "panel". The panel extends from the blade root to the blade tip and is convected agewise using a rigid wake model. The far inboard trailed vortex is placed in the radial center of the panel and the far inboard shed vortex is placed on the azimuthal center of the panel. The modeling of each of these vortices is similar to the modeling of the low resolution inboard far wake.

The tip vortex for all but the reference blade extends agewise from the blade tip to the number of wake spirals specified (see namelist NLWAKE) in the low resolution portion of the code. For the reference blade, the tip vortex and the inboard far wake begin at an agewise location determined by the parameter KNWINT specified in namelist NLHIRES.

3.3.3 Vortex Segment Location

The locations of the tip and inboard vortices must be determined at azimuthal stations in between the known low resolution results. First, the far inboard wake is linearly interpolated to the current azimuthal location, ψ, between two known low resolution stations, ψ_{lo} and ψ_{hi}. For example, Figure 3.2 shows two known low resolution far wake "panels" (which are normally represented in CAMRAD.Mod1 as two vortex elements with large core sizes) at the azimuth locations, ψ_{hi} and ψ_{lo}. The solid lines represent two consecutive known low resolution azimuth stations. Note that for each low resolution location, the wake age (ϕ) starts at zero. The dotted lines represent the current high resolution azimuthal location where information is unknown. At the current azimuth, ψ, the vortex location is determined by linearly interpolating between two points of equal wake age, ϕ. For example, the new vortex endpoint coordinates (labeled A) are determined

by interpolating between the endpoint coordinates at (ψ_{hi}, $\phi = 0^\circ$) and (ψ_{lo}, $\phi = 0^\circ$). All wake endpoint locations of the inboard far wake are similarly determined.

The tip vortex in the far wake is determined by a slightly different interpolation scheme than that used for the inboard rigid far wake. In order for the tip vortex to be translated between one low resolution location and another, a scheme shown in Figure 3.3 was implemented. Instead of interpolating being two wake endpoints with the same wake age, the interpolation is done along convection lines of the tip vortex. It can be seen that in the inboard far wake interpolation scheme, the trailed and shed vortex lines are effectively "pulled" around the azimuth with the blade instead of being "deposited" in the wake. This simplistic model is acceptable for the far wake because the influence coefficients are much less sensitive to changes in inboard far wake models than to tip vortex models. However, to include more physically realistic modeling of the tip vortex problem, the tip vortex is assumed to be "deposited" by the blade into the wake. To accomplish this, the tip vortex endpoints are interpolated along straight convection lines between the low resolution tip vortex endpoints. For example, in Figure 3.3, as in Figure 3.2, the solid lines represent known low resolution azimuth locations and the dotted lines represent the current high resolution azimuth location desired. Here, the coordinates of the tip vortex are obtained by interpolating between the known tip vortex endpoint coordinates at the locations (ψ_{hi}, $\phi = 10^\circ$) and (ψ_{lo}, $\phi = 0^\circ$). This scheme effectively translates, or convects, the vortex from one low resolution endpoint location to the next. Note that with this scheme, an additional tip vortex segment (from the blade tip to point A) must be included to connect the interpolated endpoint at location A to the tip of the blade at the location ψ. This additional vortex segment is included in the analysis.

This modeling of the far wake and tip vortex is carried out for all blades. The far wake of the reference blade begins at the wake age specified by the parameter KNWINT in namelist NLHIRES. As a historical note, KNWINT could be set equal to zero to recover an earlier version of the high resolution calculation procedure which included only the tip vortex and the inboard far wake. Since the tip vortex and inboard far wake in this case begin at the reference blade, no near wake model would be required. However, this is a crude approximation to the near wake.

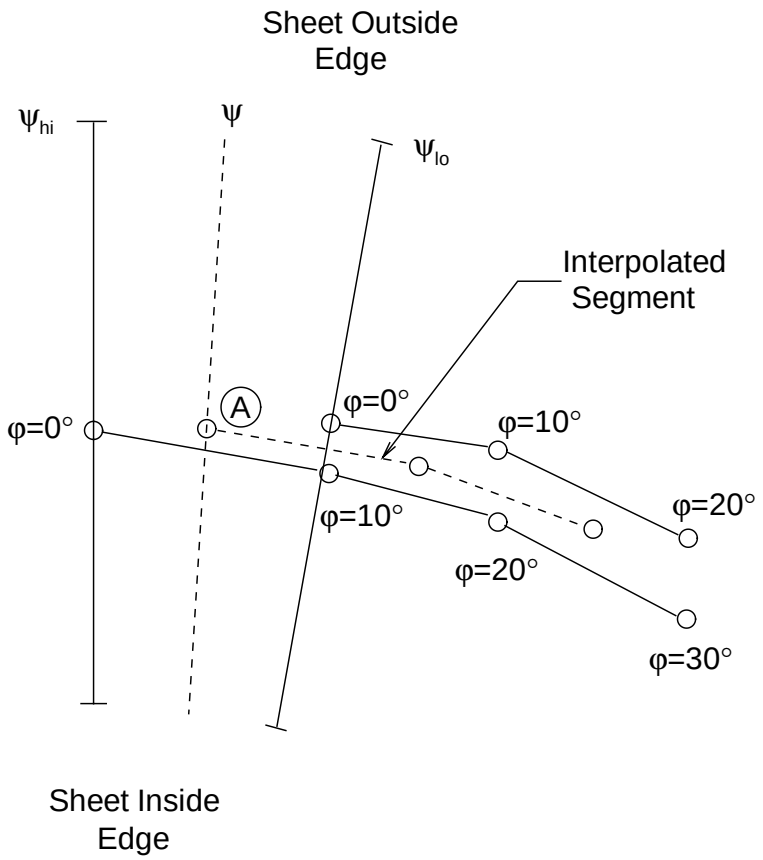


Figure 3.2: Far wake inboard geometry interpolation illustration.

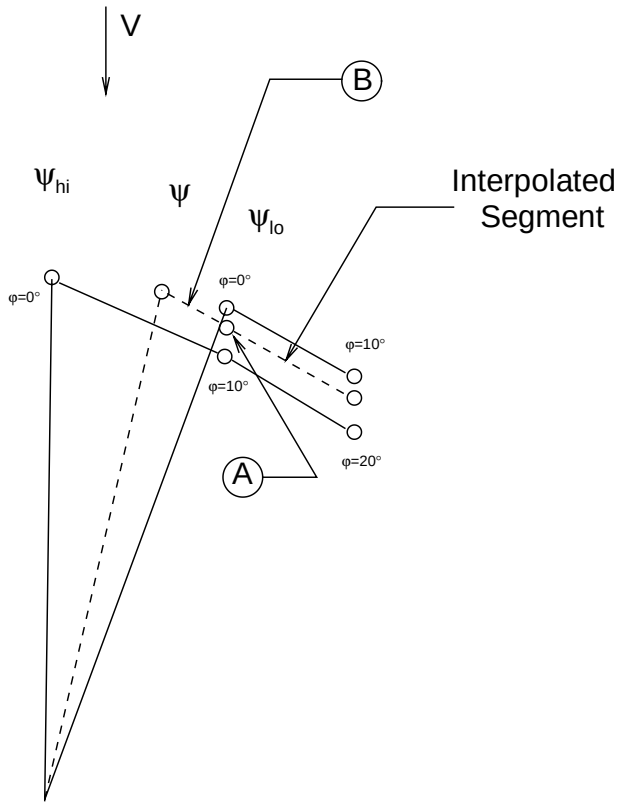


Figure 3.3: Tip vortex geometry interpolation illustration.

3.3.4 Tunnel/Fuselage Corrections

The tunnel/fuselage correction model is included in determining the tip vortex location at a high resolution. The usage of this model is similar to the low resolution application that is described in Chapter 2 and no new theory is presented here. In the high resolution scheme, the tunnel/fuselage additional wake distortions are added to each low resolution wake endpoint coordinate before interpolation is made to the current location. Since the addition of the tunnel/fuselage components of the additional wake distortion takes place before the interpolation, the wake geometry components of the tunnel/fuselage correction model do not need to be intepolated separately to a high resolution. However, the additional velocities over the rotor disk are linearly interpolated immediately after input for use in the high resolution portion of CAMRAD.Mod1. All of these processes are internal to CAMRAD.Mod1 and no additional user intervention is required.

3.3.5 Rollup Model

The rollup model modifications to the low resolution portion of CAMRAD.Mod1 were in discussed in Chapter 2. The high resolution implementation of the rollup model is only an extension of the low resolution implementation. No new theory is necessary. As implemented, the rolled-up tip and secondary vortex locations are added to the low resolution wake endpoint locations (as was described for the tunnel/fuselage corrections above) before interpolating to the current azimuth angle and wake age in HIRES. If the default option for the multi-core model is being used for the tip and secondary vortices (*i.e.,* the array of constant size cores), then the low resolution multi-core vortex core sizes are used at the low resolution wake endpoints as is done for the vortex locations described above. However, if the "single, variable core, multi-core model" or if the "variable multi-core, multi-core model" is being used, the low resolution results for these internally calculated core sizes are interpolated to a high resolution in the subroutines ROLLUPH1 and ROLLUPH2 (for rotor-1 and rotor-2, respectively). These new interpolated arrays are then used in the calculation of the high resolution wake influence coefficients. Again, all of these interpolations and connections are made internally in CAMRAD.Mod1 and no other user intervention is required.

3.3.6 Vortex Segmentation

In determining the wake influence coefficients of the high resolution tip vortices, it was discovered that under certain circumstances, the tip vortex endpoints, being connected by straight line segments, aligned such that the junctions where the vortex endpoints are connected produced an artificial unsteady effect in the aerodynamic loading. These artificial effects were reduced by decomposing the offending vortex segments into several new segments (see Figure 3.4). To accomplish the decomposition, a criterion was set such that if a tip vortex segment is within a 10 $^\circ$ azimuth angle of the reference blade, then that vortex segment along with the two vortex segments on either side of it are subdivided into five segments. The resulting six vortex endpoint coordinates and strengths are linearly interpolated from the original four endpoint coordinates and strengths by the following formulae:

$$q(1) = Q(1) \tag{3.1}$$

$$q(2) = \frac{1}{3}Q(1) + \frac{2}{3}Q(2) \tag{3.2}$$

$$q(3) = \frac{2}{3}Q(2) + \frac{1}{3}Q(3) \tag{3.3}$$

$$q(4) = \frac{1}{3}Q(2) + \frac{2}{3}Q(3) \tag{3.4}$$

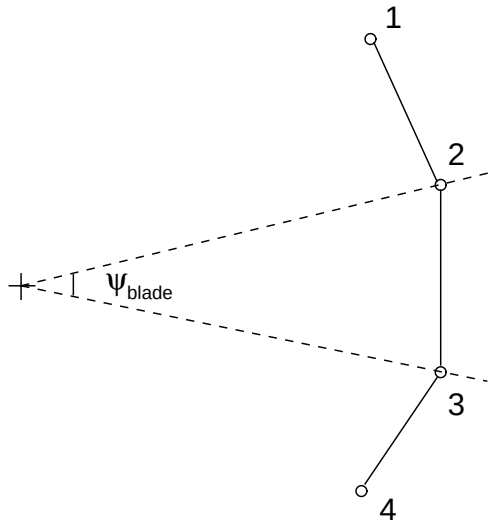
$$q(5) = \frac{2}{3}Q(3) + \frac{1}{3}Q(4) \tag{3.5}$$

$$q(6) = Q(4) \tag{3.6}$$

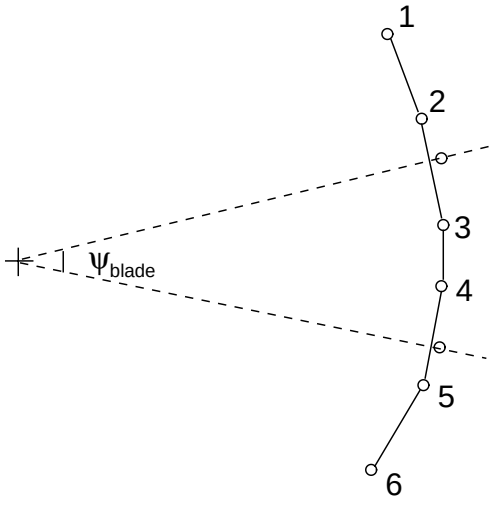
where q is the new divided segment position or strength and Q is the original position or strength. This vortex segmentation is controlled by the parameter OPSEGD in namelist NLHIRES. If OPSEGD = 0, no segmentation is performed. If OPSEGD = 1, segmentation is performed.

3.3.7 Aerodynamic Collocation Point Shifts

The aerodynamic effects of a swept planform in the high resolution solution procedure is addressed. With a swept planform (*i.e.,* sweep of the quarter-chord line of the blade), the aerodynamic collocation points are spatially displaced. Thus, a position change is made to the collocation point coordinates in the influence coefficients calculation (see Figure 3.5).



w/o Vortex Segmentation



w/ Vortex Segmentation

Figure 3.4: Illustration of vortex segmentation.

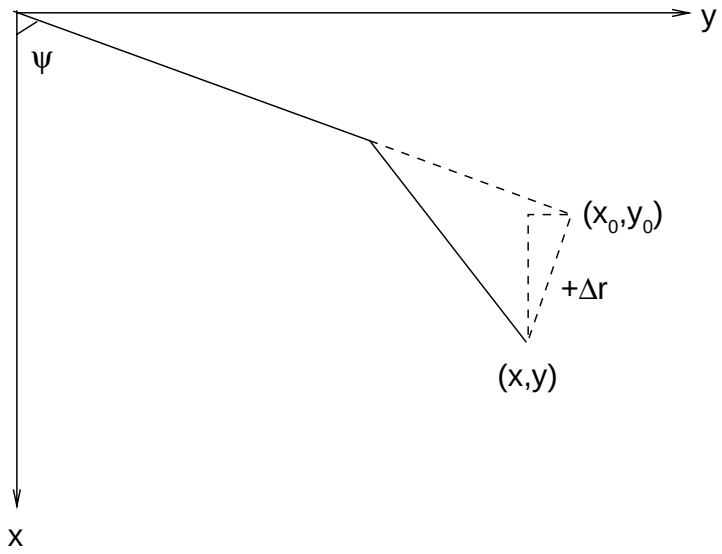


Figure 3.5: Swept planform collocation point shifts.

No modifications were made to alter the location of the tip vortex due to blade tip position changes nor were modifications made to change the structural model of the blade. Since the tip vortex location is unmodified by the collocation point shift, care must be used to make sure that the far wake of the reference blade begins at a sufficient wake agewise distance from the reference blade. The definition of sufficient depends on the rotor being analyzed. For example, if the far wake were to start at the reference blade, an aft shifted collocation point would "see" tip vortex elements ahead of the aft shifted blade tip since the vortex begins at the original (unshifted) reference blade tip. Normally, the far wake does not extend up to the reference blade tip (*i.e.,* OPNEGV = 1, or KNWINT > 0).

If the internal high resolution vortex lattice near wake is used with the shifted collocation points, however, the shifted location problem just discussed is not an issue during the lattice near wake calculations. This is because the near wake lattice model geometry is determined by the blade geometry. The shifted collocation point method is not included in the low resolution portion of the code.

The shift in collocation points is a user specified shift in the lead-lag direction (parallel to the hub plane) and does not include a vertical shift. The parameters DRPROOT, DRPTIP, and DRP. These parameters are found in namelist NLHIRES. The meaning of these variables are that (1) DRPROOT is the shift of the blade root (positive aft), (2) DRPTIP is the shift of the blade tip (positive aft), (3) DRP(100) is the shift of the high resolution collocation points (positive aft).
The equations used to determine the new collocation point coordinates are as follows (see Figure 3.5):

$$x = x_0 + \Delta\bar{r}\sin\psi \tag{3.7}$$

$$y = y_0 - \Delta\bar{r}\cos\psi \tag{3.8}$$

$$z = z_0 \tag{3.9}$$

where $\Delta\bar{r}$ is DRPROOT, DRPTIP, or DRP depending on the current location on the blade.

3.3.8 Far Wake Loading

Once the wake influence coefficients are known, they are assumed to be invariant throughout the subsequent loading calculations. This is similar to that found in the low resolution portion of CAMRAD.Mod1. As was

seen in Figure 3.1, after the influence coefficients calculation in subroutine WKC1INT, subroutine MTNRINT1 is called to calculate the aerodynamic loading. This subroutine is a simplified version of the low resolution subroutine MOTNR1. The simplifications are possible since MTNRINT1 is not required to be involved in a trim loop. Its only functions are to call the blade position subroutine MTNBINT1 (the high resolution equivalent to the low resolution subroutine MOTNB1), call the airloads calculation subroutine AEF1INT (the high resolution version of the low resolution subroutine AEROF1), and to print out the results to a file. The file write parallels the low resolution output file and is described in the following subsection.

If deemed necessary by the user, the far wake loading information calculated may be used to calculate a new circulation distribution which in turn could be used to recalculate the induced far wake velocity. This loop may be seen in Figure 3.1 as the return arrow from below MTNRINT1 to above CIRCCAL1. Normally, this return path is not used.

3.3.9 Output Aerodynamic Information

A high resolution aerodynamic information file is output for each rotor in the same format as described for the low resolution aerodynamic information file described in Chapter 2 with the following exceptions:

1. The number of radial stations is MRAINT,
2. The number of azimuth stations is MPSIINT,
3. There are three comment lines at the top instead of two,
4. The moment coefficient is excluded,
5. The blade flapping deflection is excluded,
6. The interpolated maximum bound circulation is the value interpolated from the low resolution solution,
7. The value of maximum bound circulation after the high resolution calculation is included.

The unit numbers for the far wake aerodynamic files for rotor-1 and rotor-2 are 17 and 57, respectively.

3.3.10 Output Induced Velocity Information

For each rotor, the far wake induced velocity is written to a file (in the subroutine VINDCAL1) at the end of the far wake phase of the high resolution calculation. The file contains all three components of the induced velocity for all blade sections at all azimuth stations in the following format:

```
    IF(IWR.EQ.ITERINT)WRITE(18,830)
830 FORMAT(1X,'INDUCED VELOCITY (ROTOR 1):')
    DO 850 I=1,3
    DO 850 JJJ=JFIRST,JLAST
850 IF (IWR.EQ.ITERINT) WRITE(18,860)
   1                    (VINDINT(I,JR,JJJ),JR=1,MRAINT)
860 FORMAT(10F12.6)
```

These velocity files are written to unit numbers 18 and 58 for rotor-1 and rotor-2, respectively.

3.4 High Resolution Lattice Near Wake Model

3.4.1 Introduction

As discussed in the previous section, the far wake consists of the bound vortices of all blades except the reference blade, the tip vortices of all blades, and an inboard rigid wake of all blades. The far wake does not start immediately from the blade but is offset in wake age by the value KNWINT (see namelist NLHIRES). From the blade to the first far wake elements, a near wake model is employed. The near wake model can be viewed as a transfer function as shown in Figure 3.6 and the far wake aerodynamic loading can be viewed as a forcing function for the near wake model. The near wake transfer function has been formulated by two separate means: (1) a vortex lattice model, which uses, among other things, the far wake aerodynamic loading (circulation) information, and (2) an external "Indicial Post-Processor" (IPP) code, which uses the total far wake velocities, the wake induced velocity, and the blade pitch at all blade radial and azimuthal stations. The vortex lattice near wake model is internal to the high resolution part of CAMRAD.Mod1 and is discussed in this section. The IPP code is external to the CAMRAD.Mod1 code (but is still part of the system called "HIRES"), and is discussed in a later chapter. Either of the two near

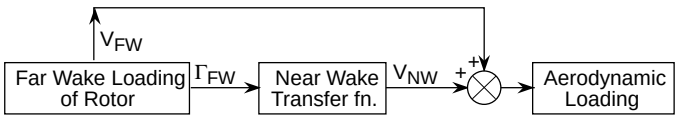


Figure 3.6: Schematic of near wake transfer function.

wake models may chosen, but not both; however, note that the lattice near wake model has not been exercised or validated.

In this section, the near wake lattice model is presented. The model uses as input, some portion of the far wake information and as an output a near wake induced velocity which when combined with the far wake induced velocity may be used to calculate the aerodynamic loading on the reference blade. The lattice near wake loop involves calculating influence coefficients of all wake elements in the near wake lattice, calculating the velocity induced at blade collocation points by these vortex elements, and calculating the total aerodynamic loading (and circulation) at the blade collocation points with the newly calculated near wake induced velocities and previously calculated far wake induced velocities. Each aspect of this model is discussed in detail.

3.4.2 Lattice Geometry

The high resolution near wake lattice is comprised of vortex elements placed on wake "panels" behind the reference blade. These panels extend in a wake agewise direction which is perpendicular to the reference line (the unswept, straight quarter-chord) of the reference blade, as shown in Figure 3.7. The "side edges" of the panels are defined by the location of the collocation points on the reference blade. The edges of the panels in the radial direction are determined by the radial shape of the blade. Therefore the panel shape is always determined by the blade shape (radially, vertically, and azimuthally). So, if the collocation points are shifted due to sweep, as discussed earlier, the near wake is adjusted so that it is always attached to the blade. The equations used to determine the near wake panel edge locations, x_{nw}, y_{nw}, and z_{nw}, associated with each radial station, i, are as follows:

$$x_{nw}(i) = r_{nw}(i)\cos(\psi - \phi) \tag{3.10}$$

$$y_{nw}(i) = r_{nw}(i)\sin(\psi - \phi) \tag{3.11}$$

$$z_{nw}(i) = z_b(r, \psi - \phi) \tag{3.12}$$

$$r_{nw}(i) = \sqrt{x_b^2 + y_b^2 + z_b^2} \tag{3.13}$$

where x_b, y_b, and z_b are the coordinates of the blade at the i-th radial station, ψ is the reference blade azimuth location, and ϕ is the wake age. This calculation is performed for all wake elements of the ages $\phi = 0$ to KNWINT*$\Delta\psi$.

With the panel edge locations and therefore the panel corner locations known, the vortex elements representing the panel may be located. Each panel's trailed vortex is located in the spanwise center of the panel. That is, the trailed vortex on each panel is located by connecting the two midpoints of the leading edge of the panel and the trailing edge of the panel (see Figure 3.8). The shed vortex on each panel is found by determining the fore-aft midpoint on each panel side edge. The shed vortex on each panel is located by connecting the two midpoints just determined, then shifting the shed vortex aft by a distance of one quarter chord (c/4). At the root and tip, the trailed elements are placed in the middle panel, as for all the others, with the outside edge being defined by the end of the blade (whether root or tip). The equations used in determining the vortex locations are as follows:

$$\vec{r}_A = \frac{(\vec{r}_2 + \vec{r}_4)}{2} \tag{3.14}$$

$$\vec{r}_B = \frac{(\vec{r}_1 + \vec{r}_3)}{2} \tag{3.15}$$

$$\vec{r}_C = \frac{(\vec{r}_3 + \vec{r}_4)}{2} + \frac{(\vec{r}_4 - \vec{r}_3) * d_i}{|\vec{r}_4 - \vec{r}_3|} \tag{3.16}$$

$$\vec{r}_D = \frac{(\vec{r}_2 + \vec{r}_1)}{2} + \frac{(\vec{r}_2 - \vec{r}_1) * d_{i+1}}{|\vec{r}_2 - \vec{r}_1|} \tag{3.17}$$

where $\vec{r}$ are position vectors to the locations denoted by the subscript, and d_i is the local chord divided by four.

3.4.3 Vortex Strengths

The strength of each vortex element (trailed and shed) must be determined. The trailed vortex strength is calculated from the radial derivative of circulation and the shed vortex strength is calculated from the azimuthal derivative in circulation. The change in circulation is found by differencing the bound circulation across a particular panel. For example, the bound circulation at each point A, B, C, and D are found by the following:

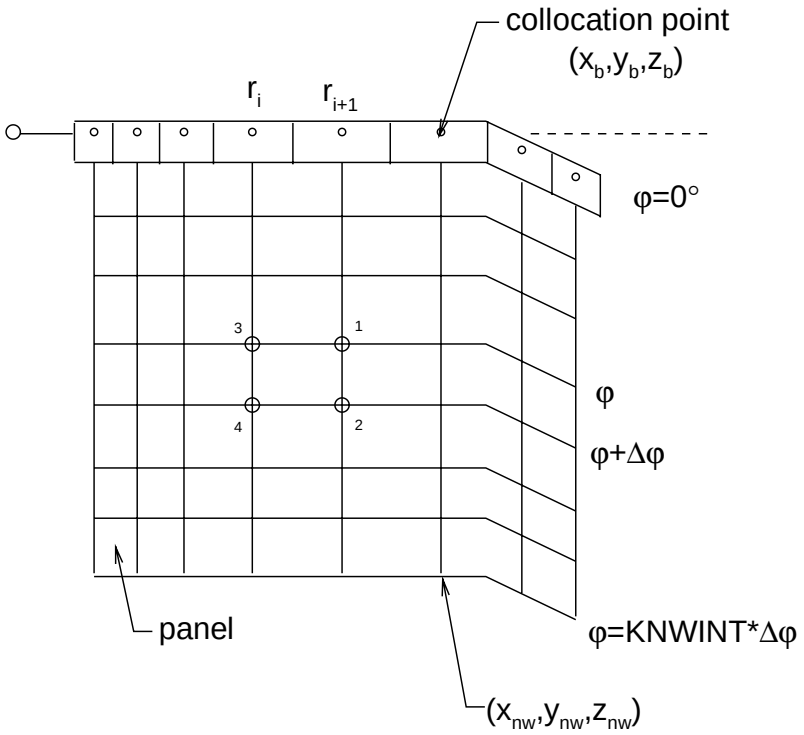


Figure 3.7: Near wake panel geometry with a swept planform.

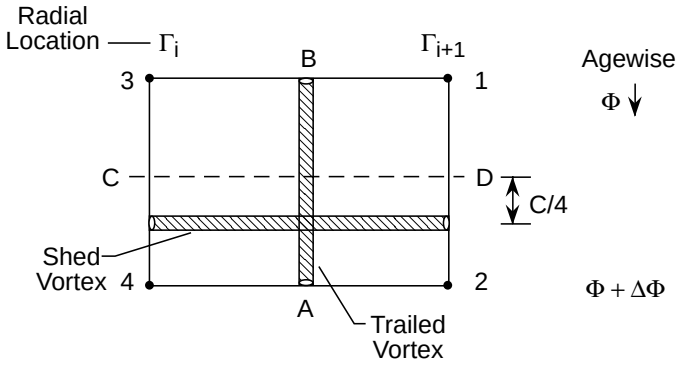


Figure 3.8: Close-up of near wake panel geometry.

$$\gamma_A = \gamma_2 - \gamma_4 \tag{3.18}$$

$$\gamma_B = \gamma_1 - \gamma_3 \tag{3.19}$$

$$\gamma_C = \gamma_4 - \gamma_3 \tag{3.20}$$

$$\gamma_D = \gamma_2 - \gamma_1 \tag{3.21}$$

where $\gamma_1, \gamma_2, \gamma_3$, and γ_4 are the circulations at the corners of the panels. These circulations are equal to the blade bound circulation at the azimuth for which the vortex endpoint was "released" from the blade. The circulation distribution between points A and B and between points C and D may optionally be a stepped distribution with a step in circulation at the panel midpoint or may be instead a linear circulation distribution. The vortex core model may also be either a distributed core model or a concentrated core model. These options are controlled by the namelist NLHIRES variables MDLSNW, MDLTNW, OPCSNW, and OPCTNW. The core size of the trailed and shed vortex elements may be an input constant for each direction, or may be calculated internally as controlled by the NLHIRES variables CORETNW and CORESNW.

3.4.4 Total Loading

With the trailed and shed wake geometry known, the vortex core size known, and the type of core (none, concentrated, or distributed) known, the influence coefficients of the trailed and shed elements on a particular blade

collocation point may be found. The influence coefficients are then multiplied by the appropriate circulation, as determined above, to obtain the velocity contribution from that vortex element acting at the blade collocation point. As the procedure continues, all velocity contributions of all near wake vortex elements for a blade at a fixed azimuth are determined. These velocities are added to the appropriate far wake velocities to determine a total velocity at the blade collocation point due to all wake elements. Once the total velocity is known for a blade at the current azimuth, the aerodynamics at that azimuth are determined. The azimuth is incremented and the process is repeated. Normally, it is expected to take more than one revolution (ITERNW > 1) to converge the near wake solution since during the first revolution (iteration), the starting transients for the near wake solution must be allowed to decay to an acceptable level.

3.4.5 Option to Reduce Number of Panels

NLHIRES contains several near wake option parameters. There is an option to reduce the number of panels in the near wake as a predetermined function of the wake age, ϕ. In principal, this technique should reduce the computation time in the lattice near wake model. By default, the parameter, OPNWMIN $= 0$, which suppresses the reduction in the number of panels. If OPNWMIN =1, the number of panels is reduced by a factor of two every 5° of wake age. (Actually, the number of panels is exactly one-half the previous number if the previous number of panels is even and the number is one half of the previous number of panels + 1 if the number of panels is odd.) For example, if there are 80 panels at the wake age $\phi = 0$ degrees, then starting at $\phi = 5^\circ$ there will be 40 panels, and at $\phi = 10^\circ$ there will be 20 panels, *etc.* The reduction in panel number will continue until either the minimum of two panels is reached or until the end of the near wake region is reached, whichever comes first.

3.4.6 Circulation Update Option

There exists an option to update the circulation being used with the wake influence coefficients to calculate near wake induced velocities at blade collocation points. The NLHIRES parameter OPNWCRC is used to control this updating. The updating may be performed at the end of each near wake iteration (OPNWCRC $= 1$) or may be updated at each azimuth step as it is calculated (OPNWCRC $= 0$). The default is to update at each azimuth

step as the circulation is calculated. This default should reduce the number of near wake iterations required.

3.4.7 Output Information

Output from the lattice near wake section of the high resolution calculation is similar to the outputs from the low resolution calculation and from the far wake calculation. Again, there is one file output for each rotor. The unit numbers for rotor-1 and rotor-2 are 19 and 59, respectively. The format of each file is the same as the far wake file described earlier with the following exceptions:

1. The vertical component (z coordinate) of total induced velocity is included as an additional entry after the MAXIMUM BOUND CIRCULATION - CIRCN values,

2. The vertical component (z component) of the high resolution near wake induced velocity is included as an additional entry at the end of the file.

The formats of the above items are as follows:

```
     WRITE(19,895)
895  FORMAT(1X,'INDUCED VELOCITY - Z COMPONENT')
     DO 896 I=1,MRAINT
896  WRITE(19,700)(VINDTOT(3,I,J),J=JFIRST,JLAST)
     WRITE(19,897)
897  FORMAT(1X,'NEAR WAKE INDUCED VELOCITY - Z COMPONENT')
     DO 898 I=1,MRAINT
898  WRITE(19,700)(VNW(3,I,J),J=JFIRST,JLAST)
```

3.4.8 Known Caveats

It should be noted that the high resolution lattice near wake model discussed in this section has not been exercised or validated.

Chapter 4

Indicial Post-Processor

4.1 Indicial Post-Processor

4.1.1 Introduction

As discussed in previous chapters, conceptually, the user has an option as to which high resolution near wake model to use: (1) the lattice near wake model internal to the HIRES code, or (2) the Indicial Post-Processor. Chapter 3 discusses the high resolution lattice near wake model that is internal to the HIRES portion of CAMRAD.Mod1. This chapter discusses the Indicial Post-Processor (IPP). The IPP is a stand-alone code that uses output information from HIRES (from the "Far Wake Loop" only). This far wake information from HIRES is combined with an indicial aerodynamics method for the near wake to calculate the total aerodynamics on the reference blade at a high azimuthal and radial resolution. The IPP uses a combination of techniques presented in several publications by T.S. Beddoes and G. Leishman (Ref. [13, 14, 15, 16]).

4.1.2 Solution Procedure

Conceptually, the IPP calculates the unsteady aerodynamic loading at each high resolution collocation point on the reference blade using the velocities at each collocation point (as determined in the Far Wake Loop of HIRES) as a gust field. That is, in the Far Wake Loop of HIRES, velocities due to blade rotation, velocities due to blade motion, and velocities induced by the "far-wake" vortex system are computed. The IPP then calculates the unsteady aerodynamic loading (response) on the reference blade as if there

were a gust field (given by the velocities from the Far Wake Loop of HIRES) traveling past the stationary reference blade.

In the indicial aerodynamics method, the high resolution near wake is divided into two parts: (1) the shed near wake, and (2) the trailed near wake. (Note that dividing the near wake in this manner is a common practice (Ref. [11]).) Each of these components of the near wake (shed and trailed) are addressed in the IPP. That is, the IPP combines the far wake velocities from HIRES with the velocities from the near wake trailed vortex system model, and uses those velocities to calculate a quasi-steady angle of attack and a Mach number. The indicial formulations of Beddoes and Leishman are then used to relate the quasi-steady angle of attack and Mach number at a blade section to a blade sectional loading. The indicial formulations essentially are empirical "curve-fits" representing the theoretical and experimental relations between quasi-steady angles of attack and Mach numbers. This chapter discusses in detail how the HIRES far wake information is "processed" by the IPP to produce high resolution loading.

4.1.3 Conceptual Program Outline

Figures 1.1 and 1.3 illustrate the computational flow of the CAMRAD.Mod1 code. It is seen (as described in earlier chapters) that the low resolution (azimuthal and radial) CAMRAD.Mod1 is executed first, followed by the HIRES portion of the code. Once these two steps are complete, the user may choose to execute the IPP to obtain high resolution loading at the high resolution collocation points on the reference blade. Note that the IPP is a true post-processor; that is, it is a stand-alone code. Even so, it is still considered part of the CAMRAD.Mod1/HIRES code system.

Figure 4.1 expands the IPP box of Figures 1.1 and 1.2 so that the IPP may be outlined in a conceptual manner. It is seen that there are three inputs to the IPP. The first input is a namelist called INLST. This namelist is used to set several parameters used in the IPP. This namelist will be discussed later. Second, the far wake information from HIRES is input. This information actually consists of two data files: (1) the far wake data file, and (2) the far wake induced velocity file. These two files will be discussed later. Third, the binary airfoil table file from CAMRAD.Mod1 is input. This input, too, will be discussed later.

The IPP consists of two "loops". The outer loop is a loop over all of the high resolution azimuth stations. At each azimuth inside the outer loop, there is a loop over all of the collocation points on the reference blade. As

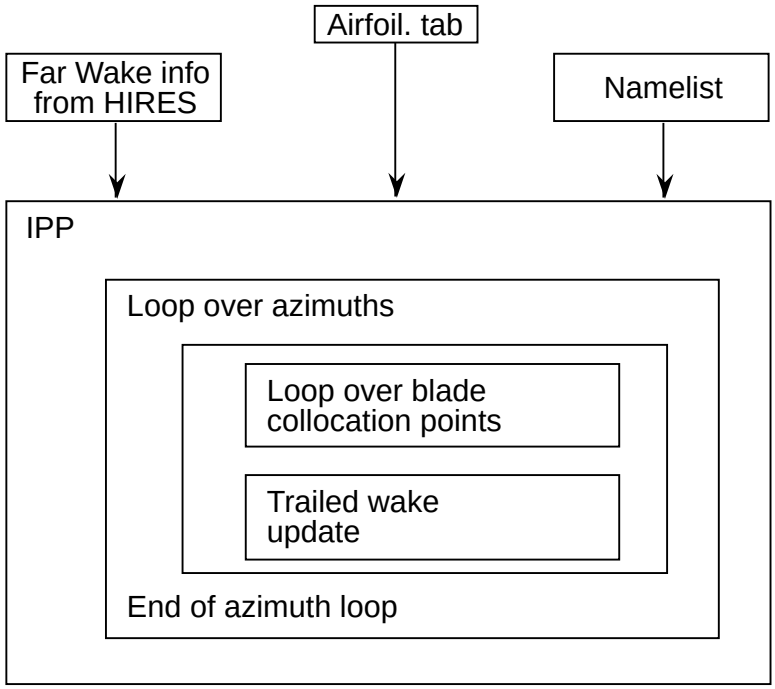


Figure 4.1: Schematic of IPP computational loops and inputs.

discussed earlier, the IPP calculates the loading on the reference blade as if it were traversing a gust field given by the far wake information from HIRES. Therefore, the reference blade is the only blade considered in the IPP. The loop over the collocation points on the blade is where the sectional blade loading is calculated. Once the loading is known on the entire span at the current azimuth station, a Trailed Wake Algorithm (TWA) is used to calculate the influence of the trailed near wake system. It should be noted that the influence of the trailed near wake system needs to be known before the blade loading can be calculated. But, since the TWA calculates the trailed near wake influence based on the blade loading, an interation is required. The iteration technique implemented is to lag the TWA calculation behind the blade loading calculation by one azimuth station. Thus, once the blade loading has been calculated in the loop over all blade collocation points, the TWA calculates the influence of the trailed near wake system, then uses that information at the following azimuth station.

4.1.4 Actual Program Outline

Figure 4.2 shows the outline of subroutines in INDICIAL, which is the executive program of the IPP. The first four subroutines are basically initialization subroutines, that will be discussed later. The last subroutine in Figure 4.2 (CLCALC) is where the "loop over all of the high resolution

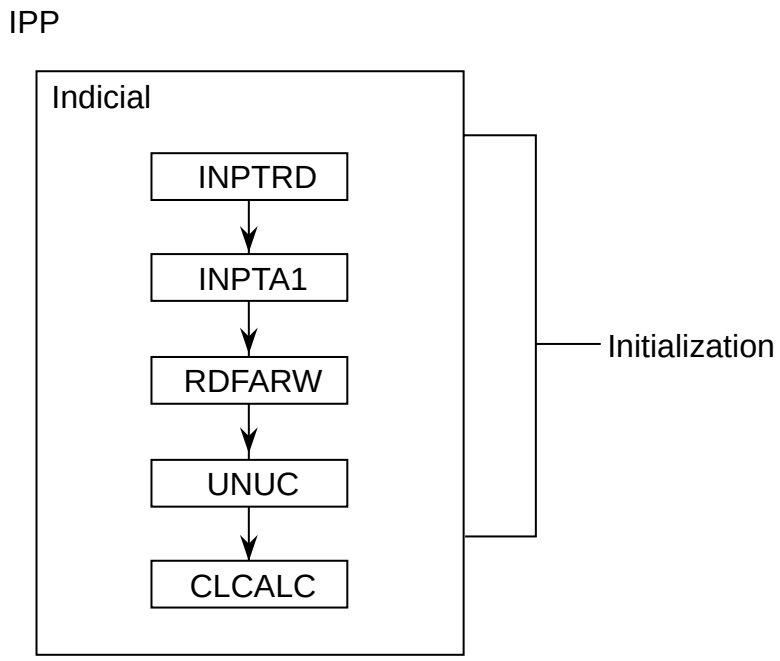


Figure 4.2: Major subroutines in the IPP.

azimuth stations" (see Figure 4.1) is located. This subroutine will also be discussed later. The first stage in the execution of the IPP is intitialization of parameters. This will now be examined by examining each of the initialization subroutines individually.

4.1.5 Subroutine INPTRD

The first subroutine in the IPP, called INPTRD, initializes several namelist input parameters to default values and reads the input namelist, INLST. These input parameters are user defined quantities and are case dependent. Each of the namelist input variables are discussed later (along with their default values). The subroutine INPTRD also checks several input parameters for valid values. For example, the number of azimuth stations, NAZM, is compared to the maximum number of stations (max. = 720); the number of radial stations, NRAD, is compared to the maximum number of radial stations (max = 100); and the maximum number of revolutions in the solution (*i.e.,* the number of "outer azimuth station loops") are compared

to the maximum number of steps (max = 1440). If errors are detected during these comparisons, the code ends execution and prints an error message. Since one of the input variables is the file name of the CAMRAD.Mod1 binary airfoil table to be used for this case, the IPP next calls the subroutine INPTA1.

4.1.6 Subroutines INPTA1

One of the input parameters in the namelist INLST is the file name of the CAMRAD.Mod1 binary airfoil table to be used in this execution of the IPP. To read this unformatted binary airfoil table, the IPP calls the subroutine INPTA1. This subroutine is identical the the subroutine INPTA1 from CAMRAD.Mod1 and stores the necessary information in arrays in the IPP code.

4.1.7 Subroutine RDFARW

Two other input parameters are (1) the file name of the far wake information file from HIRES and (2) the file name of the far wake induced velocity file from HIRES. Both of these files are read using the subroutine RDFARW. As discussed in the "Output Aerodynamic Information" subsection of Chapter 3, there are many parameters output in the far wake information file. The information of interest in this file is (1) blade section velocities parallel to the hub plane, (2) the blade section velocities perpendicular to the hub plane, (3) the blade section radial velocity, (4) the pitch angle of the blade section, and (5) the blade radial collocation point locations. RDFARW reads the far wake information file, extracting the above five quantities while skipping over the information in the file that is not required by the IPP. After required information from the far wake information file has been extracted, the file is closed.

Next, RDFARW opens the far wake induced velocity file output from HIRES. This file contains the three components of wake induced velocity from the far wake vortex system from the Far Wake Loop of HIRES. For the IPP, only the vertical (perpendicular to the hub plane) component of this velocity is required. Thus, RDFARW extracts only the vertical component of far wake induced velocity from this file. Once that is done, the file is closed.

Finally, RDFARW converts all of the newly extracted non-dimensional quantites into dimensional quantities for use in the IPP.

4.1.8 Subroutine UNUC

The input information read from the far wake information file from HIRES is in the hub coordinate system used by CAMRAD.Mod1. For example, velocities at each blade section are given in components that are parallel and perpendicular to the hub plane. However, the IPP uses velocities that are parallel and perpendicular to the blade chord. Subroutine UNUC uses the far wake aerodynamic velocity and pitch information to calculate the velocities parallel and perpendicular to the chord of the blade section. Figure 4.3 shows the relations among the velocities U_T, U_P, U_N, and U_C. The following equations are used to calculate U_N and U_C from U_T, U_P, and V_z;

$$U_{tot} = \sqrt{U_T^2 + U_P^2} \tag{4.1}$$

$$U_{tot2} = \sqrt{U_T^2 + (U_P - V_z)^2} \tag{4.2}$$

$$U_C = U_{tot} \cos(\theta - \phi) \tag{4.3}$$

$$M = \frac{U_c}{SOUND} \tag{4.4}$$

$$U_N = U_{tot2} \sin(\theta - \phi) \tag{4.5}$$

U_T is the velocity parallel to the hub plane at the blade section, U_P is the velocity perpendicular to the hub plane at the blade section, U_{tot} is the total velocity at the blade section, U_{tot2} is the total velocity excluding the vertical component of induced velocity, V_z is the vertical component of induced velocity, U_C is the component of the total velocity parallel to the chord, U_N is the component of the total velocity (excluding the vertical component of induced velocity) perpendicular to the chord, M is the Mach number calculated from U_C, SOUND is the input speed of sound, θ is the blade pitch angle, and ϕ is the inflow angle calculated from U_T and U_P at the blade section.

Since the quantities U_T and U_P were obtained from a lifting line aerodynamics code, it is consistent to use such approximations as are introduced in the previous equations. With these approximations, Figure 4.4 illustrates the manner in which the IPP views the aerodynamic far wake environment in which the quarter chord advances in space (*i.e.,* ψ). Using this figure, the aerodynamic environment may be seen as a reference point (here, the quarter chord) traveling in a local free stream velocity, U_C, with a perpendicular velocity determined by U_N and V_z. This is equivalent to the blade

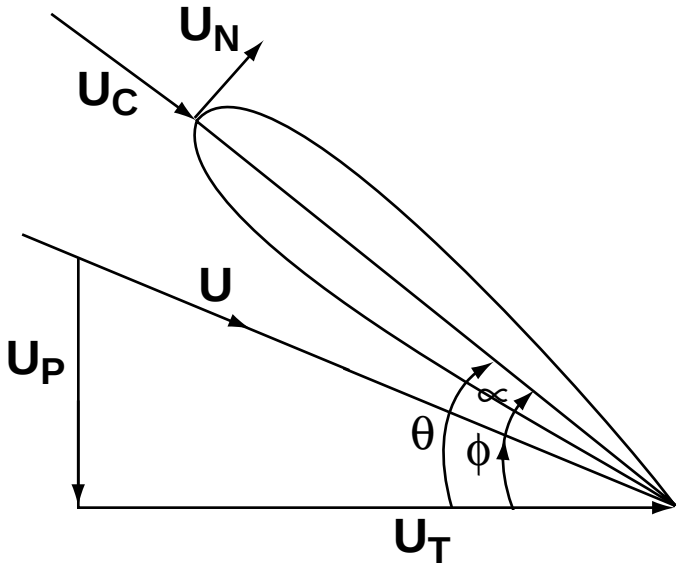


Figure 4.3: Schematic of coordinate system used in subroutine UNUC.

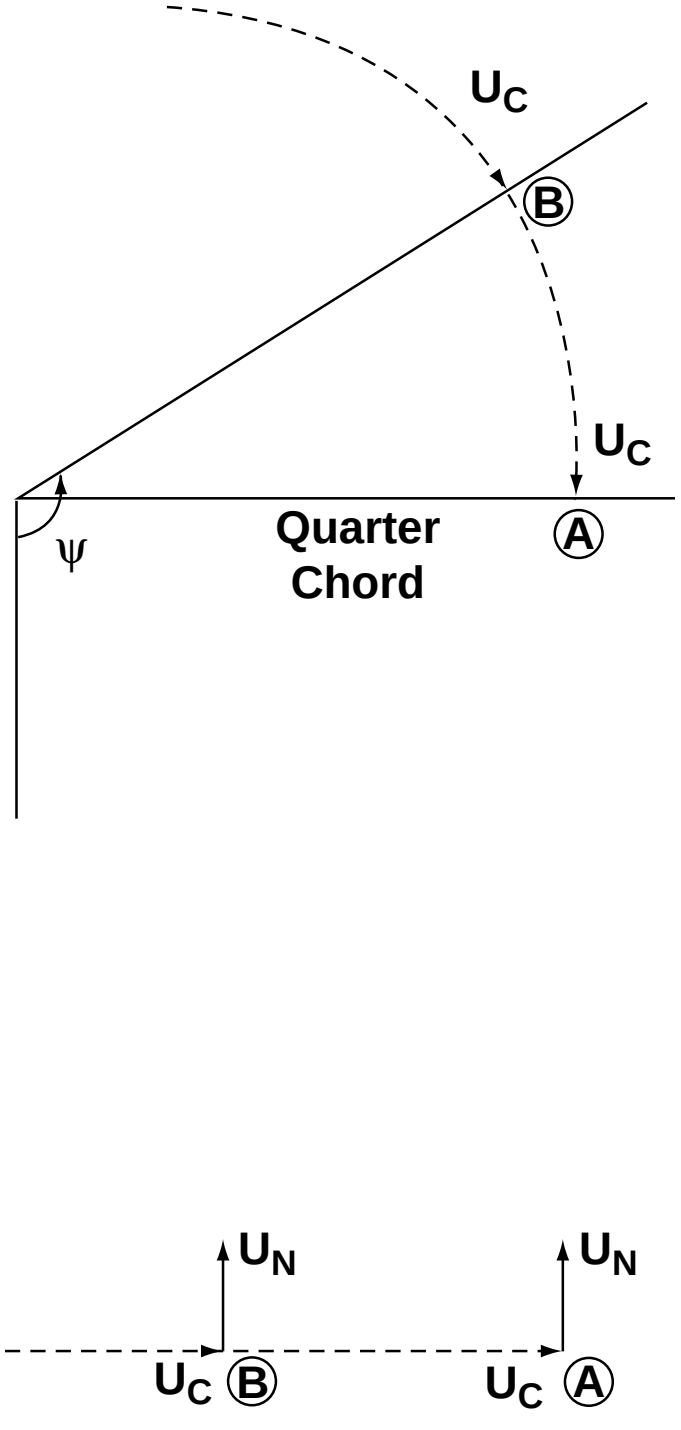


Figure 4.4: Velocities seen by blade section as it travels through gust field.

remaining stationary and allowing the far wake velocities to convect past the blade as a gust field.

An additional quantity calculated in UNUC is the pitch rate at all blade sections and azimuth locations. This is calculated using a central difference formula:

$$\dot{\theta}(i,j) = \frac{\theta(i,j+1) - \theta(i,j-1)}{2\Delta t} \tag{4.6}$$

where $\dot{\theta}$ is the pitch rate, θ is the pitch angle, index i is the radial location, index j is the azimuthal location, and Δt is the time step between azimuth locations.

4.1.9 Subroutine CLCALC - Introduction

The aerodynamic loads (forces and moments) are determined in CLCALC. These forces and moments are determined using the indicial aerodynamic functions in the mentioned References. These indicial aerodynamic functions use an effective angle of attack at a particular instant and relate it to the loads at that instant. The loading at any particular instant is split into several parts: (1) circulatory loading, (2) impulsive loading, (3) loading due to trailing edge separation, and (4) loading due to leading edge separation. The subroutine CLCALC calculates the circulatory loading effects and calls other subroutines to account for the other effects (these will be discussed later).

Conceptually, CLCALC uses the information calculated in UNUC and applies indicial aerodynamic functions to the information at each blade section and azimuth to determine the aerodynamic forces and moments (circulatory loads) there. (If the optional impulsive loads are included, CLCALC calls IMPS to calculate the impulsive loading effects.) In practice, both the circulatory and impulsive (non-circulatory) effects are two dimensional effects; the three-dimensional effects of the trailed near wake, a Trailed Wake Algorithm (TWA) similar to that of T.S. Beddoes (Ref. [13]) is used to model the effects of a trailed near wake. These two dimensional effects are split into circulatory and non-circulatory effects.

To calculate the loads (circulatory and non-circulatory), effective angles of attack are required (see mentioned References). These effective angles of attack are calculated in the subroutine INTGRL (discussed later). This subroutine returns (to CLCALC) the effective angles of attack for the circulatory lift and moment and for the non-circulatory lift and moment. How-

ever, these angles of attack calculated in INTGRL are quasi-steady angles of attack. The two dimensional indicial aerodynamic functions modify these quasi-steady angles of attack to account for unsteady effects. CLCALC accounts for the circulatory loading effects. Then, the subroutine IMPS accounts for the non-circulatory effects. Once these are known, the subroutine SEPRATE accounts for the leading edge and trailing edge separation effects.

4.1.10 Subroutine CLCALC - Coding

The outer loop in the subroutine is over the number of azimuth steps defined by the namelist input variables NAZM and NREVS. The number of steps used in the calculation is NAZM*NREVS. The maximum number of steps is 1440. At each step of the outer (ψ) loop (see Figure 4.5), all quantities are calculated at each radial station on the blade.

As stated above, the quasi-steady angles of attack are determined in INTGRL. Then, CLCALC modifies these quasi-steady angles of attack to account for unsteady circulatory effects. From the mentioned References, the following formulae are used to account for the unsteady circulatory effects in an indicial form:

$$\alpha_{eff} = \eta_L - X - Y - Z \tag{4.7}$$

$$X = X_{old} e^{-\sigma \Delta t/T_1} + A_1(\eta_L - \eta_{L_{old}}) e^{-\sigma \Delta t/2T_1} \tag{4.8}$$

$$Y = Y_{old} e^{-\sigma \Delta t/T_2} + A_2(\eta_L - \eta_{L_{old}}) e^{-\sigma \Delta t/2T_2} \tag{4.9}$$

$$Z = Z_{old} e^{-\sigma \Delta t/T_3} + A_3(\eta_L - \eta_{L_{old}}) e^{-\sigma \Delta t/2T_3} \tag{4.10}$$

$$G = G_{old} e^{-\sigma \Delta t/T_m} + A_m(\eta_m - \eta_{m_{old}}) e^{-\sigma \Delta t/2T_m} \tag{4.11}$$

$$\sigma = \frac{2U_c}{chord}(1 - M^2) \tag{4.12}$$

$$A_1 = .165 \tag{4.13}$$

$$A_2 = .335 \tag{4.14}$$

$$A_3 = .500 \tag{4.15}$$

$$A_m = 1.00 \tag{4.16}$$

$$T_1 = 20. \tag{4.17}$$

$$T_2 = 4.5 \tag{4.18}$$

$$T_3 = 1.25M \tag{4.19}$$

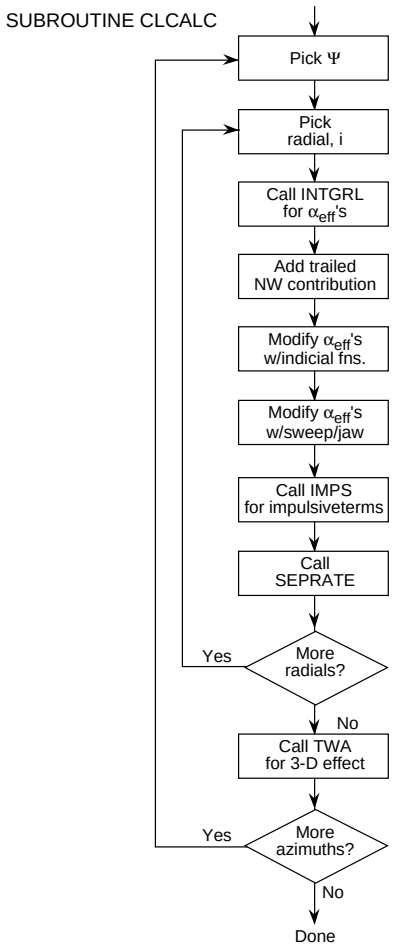


Figure 4.5: Flowchart for subroutine CLCALC.

$$T_m = M/2 \tag{4.20}$$

where η_L and η_m are the quasi-steady circulatory angles of attack calculated in the subroutine INTGRL. Once the angle of attack α_{eff} is known, it is then modified in a manner similar to that used in CAMRAD.Mod1 to model the yawed flow and blade sweep effects on the aerodynamics. Next, these modified (*e.g.,* modified by the sweep correction model) angles of attack and modified (*e.g.,* modified by the sweep correction model) Mach number are used to calculate the lift and drag coefficients C_l and C_d by airfoil table interpolation. The C_l and C_d values are used to calculate a normal force coefficient, C_n and the chordwise force coefficient, C_c. For a reference value to be used later, a $C_{n,potential}$ is calculated next by the following formula:

$$C_{n,potential} = \frac{2\pi\alpha_{eff}}{\sqrt{1-M^2}} + C_{l_0} \tag{4.21}$$

The moment coefficient, C_m, is then calculated from the indicial aerodynamics (circulatory effects). The subroutine IMPS is called next to evaluate the impulsive (non-circulatory effects) normal force, chordwise force, and moment coefficients, C_{l_i}, C_{d_i}, and C_{m_i}. The impulsive moment is then added to the C_m value. The subroutine SEPRATE is called next to calculate the effects of trailing edge and leading edge separation on the lift, drag, and moment coefficients. (The impulsive lift and drag terms are added to the previous lift and drag coefficients inside SEPRATE.) Upon return from subroutine SEPRATE, subroutine TWA is called to calculate the effects of a trailed near wake due to the current aerodynamic loading calculations.

At the end of each azimuth step in CLCALC, the following quantites may be output (depending on the input parameter ILOAD): (1) normal force and chordwise force, or (2) C_nM^2. The forces are expressed in Newtons and the C_n and M are local values. The local C_n is determined from the following definition:

$$N = \frac{1}{2}\rho(Ma)^2 cC_n \tag{4.22}$$

N is the local normal force per unit span, M is the local Mach number, c is the local chord, a is the speed of sound, C_n is the local normal force coefficient.

4.1.11 Subroutine INTGRL

As seen in Figure 4.5, the first task in CLCALC after initialization is to invoke INTGRL to calculate the effective quasi-steady angles of attack (using a weighted integral over the chord). These are later to be used in the circularory lift, circulatory moment, impulsive lift, and impulsive moment calculations. The following integrals are used to determine the effective angles of attack:

$$\eta_L = \frac{1}{\pi}\int_0^{2\pi} \frac{w(\theta)}{U_C}(1-\cos\theta)d\theta \tag{4.23}$$

$$\lambda_L = \frac{1}{2}\int_0^{2\pi} \frac{w(\theta)}{U_C}\sin^2\theta d\theta \tag{4.24}$$

$$\cos\theta = 1 - \left(\frac{x}{(c/2)}\right) \tag{4.25}$$

where η_L is the effective angle of attack at the 3/4 chord point for the circulatory terms, λ_L is the effective angle of attack for the impulsive terms, θ is the integration coordinate, c is the chord, x is the chordwise location, $w(\theta)$ is the downwash at the θ coordinate, and U_C is the chordwise velocity at the θ coordinate.

To compute these integrals, the downwash velocity $w(\theta)$ and U_C as a function of θ must be found. CAMRAD.Mod1 is not capable of providing these intantaneous velocity distributions over the chord. Thus the velocities are approximated velocities over the chord. It is assumed that the chord at any radial station "spans" several points where the velocities are known. For example, Figure 4.6 illustrates a chord at a particular radial and azimuth station. Note that the azimuthal station is defined to be the quarter-chord location of the section. In this figure, the chord is seen to "span" several other azimuth stations which are a distance $\Omega r \Delta t$ apart (where Ω is the rotor rotation rate, r is the radial station location, Δt is the time required for the rotor to travel from one high resolution azimuth station to the next). The velocities from these other azimuth stations are used in the above integrals. However, since the integration is over the entire chord, two additional points must be introduced to complete the calculations. In general, the velocities at the leading and trailing edges of the blade chord are not known. To evaluate the velocities at these points, a linear interpolation between the two surrounding points is used. Thus, the velocities at points on the chord (including the leading and trailing edges of the chord) are known.

The integrals of Equations 4.23 and 4.24 are evaluated using a trapezoidal rule which has been modified to include subintervals in the integration. A coordinate system in θ is used as shown in Figure 4.6. It has been determined through numerical experimentation that at least three points on the chord are required for the integral evaluations to be sufficiently accurate. This is sometimes a problem, particularly near the tip of the blade when the azimuthal resolution is inadequate. If there are less than three points defining the chord, the IPP issues a warning message and then stops executing. (The corrective action is to re-execute CAMRAD.Mod1 and HIRES with an increased azimuthal resolution.)

An additional non-circulatory term may optionally be employed. This term is not included in the references mention. To simulate effects of blade vortex interations (BVI), it is assumed that the impulsive character of the BVI is strong at the leading edge of the blade section, but decays toward the trailing edge. An approximation to this assumption is to allow an additional "step" function over the blade section. The magnitude of this function is scaled by the user. The extent of this function over the chord is also controlled by the user through input variables (IWT, WEIGHT, XOCOFF). The default values set in the code should be sufficient for many cases. These input variables are discussed in a later section.

4.1.12 Subroutine IMPS

The equations coded in subroutine CLCALC are the circulatory lift and moment terms. These are terms related to the circulation present on the airfoil and in the wake of the blade section. They serve to modify the quasi-steady angles of attack to account for shed wake effects. To include non-circulatory effects, another term is required; these terms are calculated in the subroutine IMPS. The equations coded in IMPS for the impulsive loading are as follows:

$$C_{l_i} = \frac{4}{M} H_n \tag{4.26}$$

$$C_{m_i} = \frac{-4}{M} H_m \tag{4.27}$$

$$C_{c_i} = \frac{2}{M} H_c \tag{4.28}$$

$$H_n = H_{n_{old}} e^{-\Delta t / T_i K_t} + (\lambda_l - \lambda_{l_n old}) e^{-\Delta t / 2 T_i K_t} \tag{4.29}$$

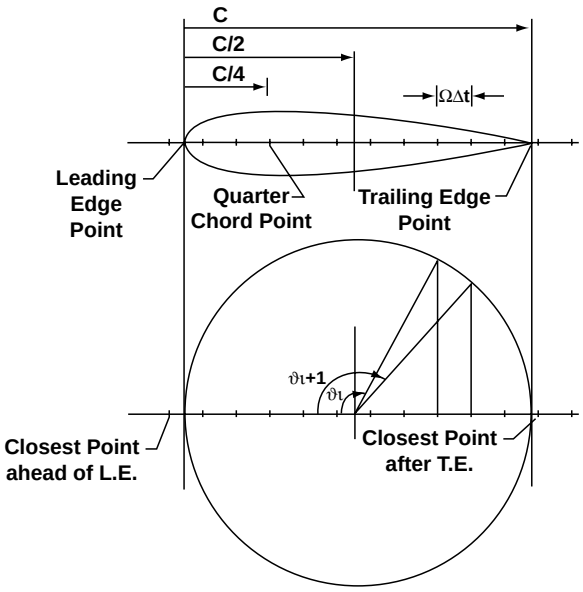


Figure 4.6: Coordinates used in the integration subroutine INTGRL.

$$H_m = H_{m_{old}} e^{-\Delta t/T_i K_t} + (\lambda_m - \lambda_{m_n old}) e^{-\Delta t/2T_i K_t} \tag{4.30}$$

$$H_c = H_{c_{old}} e^{-\Delta t/T_i K_t} + K_c (U_c - U_{c_{old}}) e^{-\Delta t/2T_i K_t} \tag{4.31}$$

$$K_c = \frac{\tau \pi}{4U_c} \tag{4.32}$$

$$T_i = \frac{c}{a} \frac{1+3M}{4} \tag{4.33}$$

$$K_i = 1 - e^{4y/c} \tag{4.34}$$

where y is the radial distance from the blade tip to the current collocation point, τ is the thickness ratio of the airfoil, a is the sound speed, and c is the chord of the current blade section. These additional terms are added to the circulatory terms discussed above.

4.1.13 Subroutine SEPRATE

Once the effective angles of attack from subroutine INTEGRL and the impulsive lift components from subroutine IMPS are known, the effects of trailing edge and leading edge separation are taken into account. This is done in the subroutine called SEPRATE. Many of the parameters involved

SEPRATE are, like the bulk of the indicial work, governed by empirical factors. This subroutine is an interpretation of the mentioned References. Since most of the empirical factors in this subroutine are functions of Mach number, the first task in SEPRATE is to linearly interpolate these parameters to the current local Mach number. Once these parameters are known, the trailing edge separation effect may be calculated. The trailing edge separation effect is discussed in detail in the mentioned References. In short, the trailing edge separation effect is a lag in the surface pressure (and thus lift) with respect to the angle of attack of the blade section due to the motion of the separation point near the trailing edge. In the references, the trailing edge separation effect is calculated from empirical formulae known *a priori* . In this application, the location of the trailing edge separation point is calculated by the following equation:

$$f'_n = \left[2 \left(\sqrt{\left| \frac{C_{n,81}}{C_{n,potential}} \right|} \right) - 1 \right]^2 \tag{4.35}$$

where f'_n is the effective separation point location, $C_{n,81}$ is the value of normal force coefficient calculated from an airfoil table lookup previously made in subroutine CLCALC (this term also includes the impulsive normal force coefficient), $C_{n,potential}$ is the value of the potential lift coefficient, also calculated previously in subroutine CLCALC. With the effective trailing edge separation point known, the motion of the separation point is lagged using equations 21 and 22 from the mentioned References. Once the lagged location of the trailing edge separation point, f'' is known, it may be reapplied to the following equation:

$$C_{n,f} = C_{n,potential} \left(\frac{1 + \sqrt{f''}}{2} \right)^2 + C_{n,i} \tag{4.36}$$

where the normal force coefficient, $C_{n,f}$, includes the effect of trailing edge separation, and the effect of the impulsive normal force, $C_{n,i}$.

The effects of leading edge separation includes the additional lift on an airfoil due to leading edge vortex separation. A simplified version of the model presented in the mentioned References is used. The simplification is that only one is allowed to be shed in the stall region. For blade vortex interaction (BVI) calculations, this is not a severe limitation as BVIs do not often appear in the stall region of the rotor. The effect of leading

edge separation on the normal force coefficient is added to the normal force contribution from the trailing edge separation.

Now, with the total normal force known, other quantities such as chordwise force coefficient and moment coefficient are calculated using the equations of the mentioned References.

4.1.14 Subroutines TWA

The subroutine TWA uses the same equations and serves the same purpose as the subroutine TWA1 in CAMRAD.Mod1 (see Chapter 2 for details).

4.1.15 Subroutine FINDNN

The subroutine FINDNN is a subroutine that determines the current azimuth index that is between one and the number of azimuth steps, NAZM, given an arbitrary azimuth index between one and NAZM*NREVS.

4.1.16 Subroutine AEROT1

Subroutine AEROT1 interpolates the input airfoil table information. It is the same as the subroutine AEROT1 from CAMRAD.Mod1.

4.2 Indicial Post-Processor Namelist Input Variables

4.2.1 Namelist INLST

The parameters in INLST are given in Table 4.1.

Table 4.1: Description of INLST Parameters

FWFILE		name of far wake file (character*80)
VINDFILE		name of far wake induced velocity file (character*80)
NRAD		number of radial stations used in calculation (max 100)
NAZM		number of azimuth stations used in calculation (max 720)
SOUND		speed of sound (m/s or ft/s)
RPM		rotor RPM
RADIUS		rotor radius (m or ft)
CHORD(NRAD)		blade chord (m or ft)
NREVS		number of rotor revolutions to calculate (default = 2)
RAE(NRAD)		edges of radial segments (usually same as high resolution variable RAEINT)
ICURV		integer parameter:
	$= 0$	use straight trailed near wake
	$= 1$	use circular arc trailed near wake (default)
EPICOR		trailed near wake core size:
	$\geq 0.$	use this constant core size for all trailed lines (re. R)
	$< 0.$	use .5*(panel width)
		(default = $-1.$)
THICK		blade thickness used in impulsive chordwise loading term (percent of chord, in decimal form)
		(default = .12)
SWP(NRAD)		sweep angle at radial segment locations (deg)
		(default = 100*0.)
DENSE		density (kg/m^3 or $slug/ft^3$)
IWT	$= 1$	use constant "step" function controlled by WEIGHT and XOCOFF (default)
	$= 2$	use new weighting function (not recommended)
WEIGHT		magnitude of impulsive BVI term (default = 1.0)
XOCOFF		x/c cuttoff location for BVI term (default = 0.25)
ILOAD	$= 1$	output normal force (default)
	$= 2$	output $C_n M^2$
IVZ	$= 0$	zero-out V_z in impulsive loading term
	$= 1$	do not zero-out V_z in impulsive loading term (default)

Chapter 5

Users' Manual: Variables and Namelists

5.1 Introduction

This users' manual is intended to contain new input variables and namelists for CAMRAD.Mod1. The intent is to document the new namelists and variables with respect to the original version of CAMRAD (*e.g.,* where the new namelists are located relative to the older namelists, *etc*). For a description of all older namelists and variables, refer to [2]. In addition, since the current version of CAMRAD.Mod1 is intended to run on a workstation with a UNIX operating (more specifically, it was set up to execute on a DEC Alpha workstation), this portion of the documentation also describes methods to execute the code in a UNIX environment (specifically, on a DEC Alpha workstation).

Even though many changes have been made internally to create CAMRAD.Mod1 from the "base" code, CAMRAD, the basic trim loop structure of the code is mostly unchanged. The changes to the loop structure that have occurred are the addition of extra loops. For example, a new "rollup-trim" loop was added to execute at the end of the original three stage wake-trim procedure. Also, both a CFD interface and a high resolution procedure (HIRES) were added to execute at the end of the entire trim procedure. For more information on these new procedures see previous chapters.

Since the goal of the CAMRAD.Mod1 code is to obtain high resolution loading information that can be used in conjunction with at helicopter noise prediction code, the remaining original procedures in CAMRAD.Mod1 (flut-

ter, flight dynamics, and transient) have not been updated, maintained, or tested. In some cases, they are not compatible with the new procedures introduced in CAMRAD.Mod1. However, if one were to execute CAMRAD.Mod1 in a manner similar to that of the original version of CAMRAD, the flutter, flight dynamics, and transient tasks should still be available - however, this has not been verified.

5.2 Summary of Job Preparation

A typical job preparation is basically unchanged from the original CAMRAD procedure. The first step is to create a BLOCKDATA file containing input information for CAMRAD.Mod1. Normally, this information is then compiled into a binary input file in a manner similar to the original version of CAMRAD. An example script for generating a binary input file is provided in the "Binary Input File Preparation" section of this manual. Next, airfoil data must be provided. This is available either by converting C81 tables into a binary airfoil table using the airfoil preparation program or by converting generic airfoil information into a binary airfoil table generated with namelist inputs to the airfoil preparation program. An example of an airfoil preparation script is given in the "Airfoil Table Preparation" section of this manual. Once these two binary files are created, the code is executed using a script file. A script template is given in the "Script Template" section.

5.3 Airfoil Table Preparation

Airfoil table preparation is normally accomplished by creating a "script" file to run the airfoil preparation program. An example script for generating the binary airfoil table to be used in CAMRAD.Mod1 from an input C81 table is as follows:

```
!/bin/csh
#
#  For Bo105 model rotor
#  NACA 23012 airfoil, standard C81 table
#
/bin/rm -rf naca23012.out
/bin/rm -rf naca23012.tab
```

```
unsetenv AFDECK1
unsetenv AFTABLE
setenv   AFDECK1 naca23012.c81
setenv   AFTABLE naca23012.tab
set campath=/usr2/boyd/Cam/Sources/Camrad_mod2_rcs
$campath/airfoil > naca23012.out <<eoj
NACA 23012 AIRFOIL     (STANDARD C81 TABLE)
 &NLTABL
    NA = 1,16,28,88,100,115,
    NM = 1,7,21,
    M  = 0.,.6,.95,
    OPREAD = 2,
 &END
eoj
exit
```

5.4 Binary Input File Preparation

```
#! /bin/csh
#
# set INPUTFILE environment variable,
# compile blockdata file,
# link files together,
# run input preparation program (input_prep)
#
#
/bin/rm -rf bo105.out
/bin/rm -rf bo105.bin
#
unset campath
unset srcpath
#
unsetenv INPUTFILE
setenv   INPUTFILE bo105.bin
set campath="/usr2/boyd/Cam/Sources/Camrad_mod2_rcs"
#
f77 -c bo105_blockfile.f
f77 $campath/blockfile.o bo105_blockfile.o
```

```
    $campath/FILEI.o -o input_prep
#
input_prep > bo105.out
exit
```

5.5 Script Template

This section describes a method to create a binary input file for CAMRAD.Mod1. First the user must create a file containing the necessary input BLOCKDATA code segment. Then, the binary input data file is created using the following example script:

```
#!/bin/csh -v
/bin/rm -f fort.* >& /dev/null
unset case
set case=runcase
unset campath
set campath=/usr2/boyd/Cam/Sources/Camrad_mod2_rcs
unsetenv INPUTFILE
unsetenv AFTABLE1
setenv   INPUTFILE bo105.bin
setenv   AFTABLE1  naca23012.tab
ln -s ${runcase}.dat             fort.7
ln -s int_${runcase}.dat         fort.17
ln -s vind_${runcase}.dat        fort.18
ln -s wake_${runcase}.dat        fort.13
ln -s blade_${runcase}.dat       fort.14
${campath}/camrad_mod1.1 >& ${runcase}.out <<eoj
 &NLCASE
   NFRS=-1,NFEIG=-1,NCASES=1,
 &END
 &NLTRIM
   VEL    =    0.15060,   CTTRIM =    0.05610,
   RPM    =   1041.000,   APITCH =      4.250,
 &END
 &NLRTR          &END
 &NLHHC          &END
 &NLHHC2         &END
 &NLHIRES
```

```
  OPINT = 1,   OPWFCOR = 0,   OPSEGD = 1,  ITERINT=1,
  MPSIINT=720,JFIRST=1 ,JLAST=720,MPSIWGP=72,
  COREINT=.01815,   KNWINT=180,   OPNEGV =1,
  WMDLINT=2,0,0,0,0,2,2,2,2,2,3,3,3,
  WKMDL1 =2,0,0,0,0,2,2,2,2,2,3,3,3,
  MRAINT=75,
  RAEINT=.1500,.1783,.1892,.2000,.2109,.2218,.2326,.2435,
         .2544,.2652,.2761,.2870,.2979,.3087,.3196,.3305,
         .3413,.3522,.3631,.3739,.3848,.3957,.4065,.4174,
         .4283,.4391,.4500,.4609,.4717,.4826,.4935,.5044,
         .5152,.5261,.5370,.5478,.5587,.5696,.5804,.5913,
         .6022,.6130,.6239,.6348,.6456,.6565,.6674,.6783,
         .6891,.7000,.7109,.7217,.7326,.7435,.7543,.7652,
         .7761,.7869,.7978,.8087,.8195,.8304,.8413,.8521,
         .8630,.8739,.8848,.8956,.9065,.9174,.9282,.9391,
         .9500,.9608,.9717,1.0000,
          0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,
          0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,
          0.,0.,0.,0.,0.,
&END
&NLBED  OPBED = 0, &END
&NLSWP             &END
&NLWAKE            &END
&NLBURST  OPBURST = 0, &END
&NLROLL
  OPROLLU=3,OPLOWR=1,OPHIWR=1,
  CORELG=.3,ITERRUP=2,ITERFRU=2,
  ITERLGC=20,FLGCORG=0.1,
  NTIPFCT=1,TIPFC0=0.,TIPFC=0.15915,0.,0.,0.,0.,0.,
                    0.,0.,0.,0.,
  NTCOR=9,TIPCORE=0.01, 0.0166, 0.0233, 0.03,
                  0.04, 0.05, 0.07, 0.1, 0.2, 0.,
  NSCOR=9,SECCORE=0.01, 0.0166, 0.0233, 0.03,
                  0.04, 0.05, 0.07, 0.1, 0.2, 0.,
  ISPIN=1,TAUC0=0.,TAUC1=1.,
  IRUZCOR=0,    OPROLSS=0,
  ICORYCB = 2,  IFWLGC = 0,   COREXP = 4,
&END
&NLMEAS IMODEIN = 0, IAEROIN = 0, &END
```

```
 &NLLOAD      &END
eoj
exit
```

5.6 Input Parameters

Following are the input parameters for CAMRAD.Mod1. Parameters that are in italic print are new parameters and/or namelists not included in the original version of CAMRAD, but are now in CAMRAD.Mod1.

Bibliography

[1] Wayne Johnson. A Comprehensive Analytical Model of Rotorcraft Aerodynamics and Dynamics, Part I: Analysis Development. NASA TM 81182, June 1980.

[2] Wayne Johnson. A Comprehensive Analytical Model of Rotorcraft Aerodynamics and Dynamics, Part II: Users' Manual. NASA TM 81183, June 1980.

[3] Brooks, T. F., Boyd, Jr., D. D., Burley, C. L., and Jolly, Jr., R. J. Aeroacoustic Codes For Rotor Harmonic and BVI Noise - CAMRAD.Mod1 /HIRES. In *2nd AIAA/CEAS Aeroacoustics Conference*, State College, PA, May 1996.

[4] Prichard, D. S., Boyd, Jr., D. D., and Burley, C. L. NASA/Langley's CFD-Based Noise Prediction System: (ROTONET/BVI) An Introduction and User's Guide. NASA TM 109147, November 1994.

[5] K. S. Brentner. Prediction of Helicopter Rotor Discrete Frequency Noise. NASA TM 87721, October 1986.

[6] Weir, D. S., Jumper, S. J., Burley, C. L., and Golub, R. A. Aircraft Noise Prediction Program Theoretical Manual - Rotorcraft Systems Noise Prediction Program (ROTONET). NASA TM 83199, 1995. Part 5.

[7] M. P. Scully. Computation of Helicopter Rotor Wake Geometry and Its Influence on Rotor Harmonic Airloads. ARSL TR 178-1, Massachusetts Institute of Technology, March 1975.

[8] Brooks, T. F., Booth, Jr., E. A., Boyd, Jr., D. D., Splettstoesser, W. R., Schultz, K. J., Kube, R., Niesl, G. H., and Streby, O. HHC -

Study in the DNW to Reduce BVI Noise - An Analysis. In *AHS/RAeS International Technical Specialists Meeting - Rotorcraft Acoustics and Fluid Dynamics*, Philadelphia, PA, October 1991.

[9] Splettsoesser, W. R., Kube, R., Seelhorst, U., Wagner, W., Boutier, A., Micheli, F., Mercker, E., and Pengel, K. Higher Harmonic Control Aeroacoustic Rotor Test (HART) - Test Documentation and Representative Results. IB 129-95/28, Deutsche Forschungsanstalt für Luft- und Raumfahrt, December 1995.

[10] Kuethe, A. M. and Chow Chuen-Yen. *Foundations of Aerodynamics: Bases of Aerodynamic Design*. John Wiley and Sons, Inc., 3rd edition, 1976.

[11] Wayne Johnson. *Helicopter Theory*. Princeton University Press, 1980.

[12] Amtec Engineering, Inc., P.O. Box 3633, Bellevue, Washington 98008-3633. *TECPLOT Version 6 User's Manual*.

[13] T. S. Beddoes. A Near Wake Dynamic Model. In *Proceedings of the AHS Specialists Meeting on Aerodynamics and Aeroacoustics*, February 1987.

[14] T. S. Beddoes. Two and Three Dimensional Indicial Methods for Rotor Dynamic Airloads. In *AHS National Specialists Meeting on Rotorcraft Dynamics*, November 1989.

[15] Beddoes, T. S. and Leishman, J. G. A Generalised Model for Airfoil Unsteady Aerodynamics Using the Indicial Method. In *42nd Annual Forum of the AHS*, June 1986.

[16] Beddoes, T. S. and Leishman, J. G. A Semi-Empirical Model for Dynamic Stall. *Journal of the American Helicopter Society*, July 1989.

[17] A. Betz. Behavior of Vortex Systems. *Zeitschrift für Angewandte Mathematik und Mechank*, 12(3), June 1932.

[18] Donaldson, C. duP. and Bilanin, A. J. Vortex Wakes of Conventional Aircraft. In *AGARDograph*, number 204. May 1975.

[19] D. B. Bliss. Prediction of Tip Vortex Self-Induced Motion Parameters in Terms of Rotor Blade Loading. In *Proceedings of the American Helicopter Society National Specialists Meeting on Aerodynamics and Aeroacoustics*, Arlington, Texas, Feburary 1987.

[20] Berezin, C. and Visintainer, J. Improved Aerodynamic Model for Wing-Body-Tail Forces in CAMRAD.Mod1. Draft Copy, 1996.

4.5 NLCASE

Namelist NLCASE

JOB		integer parameter defining job type
	$= 0$	for new job (default)
	$\neq 0$	for old job or restart job
RSWRT		integer parameter controlling restart file write
	$= 0$	to suppress write
New Jobs Only:		
NCASES		number of cases (default = 1)
BLKDAT		integer parameter defining input source:
	$= 0$	read input file (default)
	> 0	use loaded blockdata and write input file
	< 0	use loaded blockdata
RDFILE		integer parameter controlling input file read:
	$= 0$	read file for first case only
	$\neq 0$	read file for every case (default)
Old Jobs Only:		
START		integer parameter defining task:
	$= 1$	for trim restart (default)
	$= 2$	for flutter restart
	$= 3$	for flight dynamics restart
	$= 4$	for transient restart

Note that the trim restart can be followed by any or all of the other tasks (as defined by ANTYPE); for flutter, flight dynamics, or trasient restart, only that task can be done.

Additions to Namelist NLCASE

NOISFL	$= 0$	*don't perform ROTONET/WOPWOP interface calculations (default)*
	> 0	*don't perform ROTONET/WOPWOP interface calculations*

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 07704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1998	Technical Memorandum

4. TITLE AND SUBTITLE
Aeroacoustic Codes for Rotor Harmonic and BVI Noise - CAMRAD.Mod1/ HIRES: Methodology and Users' Manual

5. FUNDING NUMBERS
581-20-21-02

6. AUTHOR(S)
D. Douglas Boyd, Jr., Thomas F. Brooks, Casey Burley, J. Ralph Jolly, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23681-2199

8. PERFORMING ORGANIZATION REPORT NUMBER
L-17697

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM-1998-207640

11. SUPPLEMENTARY NOTES
D. Douglas Boyd, Jr. - Virginia Polytechnic Institute and State University, Blacksburg, Virginia
Thomas F. Brooks and Casey L. Burley - Langley Research Center, Hampton, Virginia
J. Ralph Jolly, Jr. - Jolly Development Corporation, Birmingham, Alabama

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified–Unlimited
Subject Category Category 71 Distribution: Nonstandard
Availability: NASA CASI (301) 621-0390

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***
This document details the methodology and use of the CAMRAD.Mod1/HIRES codes, which were developed at NASA Langley Research Center for the prediction of helicopter harmonic and Blade-Vortex Interaction (BVI) noise. CAMRAD.Mod1 is a substantially modified version of the performance/trim/wake code CAMRAD. High resolution blade loading is determined in post-processing by HIRES and an associated indicial aerodynamics code. Extensive capabilities of importance to noise prediction accuracy are documented, including a new multi-core tip vortex roll-up wake model, higher harmonic and individual blade control, tunnel and fuselage correction input, diagnostic blade motion input, and interfaces for acoustic and CFD aerodynamics codes. Modifications and new code capabilities are documented with examples. A users' job preparation guide and listings of variables and namelists are given.

14. SUBJECT TERMS
rotor noise, rotorcraft noise, blade-vortex interaction noise, helicopter noise prediction

15. NUMBER OF PAGES
236

16. PRICE CODE
A11

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

4.6 NLTRIM

Namelist NLTRIM

OPREAD(10)		integer vector defining namelist read structure
	= 0	to suppress read of the namelist:
		components (new job only):
		(1) NLRTR, rotor 1
		(2) NLWAKE, rotor 1
		(3) NLRTR, rotor 2
		(4) NLWAKE, rotor 2
		(5) NLBODY
		tasks:
		(6) NLLOAD, rotor 1
		(7) NLLOAD, rotor 2
		(8) NLFLUT
		(9) NLSTAB
		(10) NLTRAN
NPRNTI		integer parameter controlling input data print
	= 0	use "short form" print (no input data print)
ANTYPE(3)		integer vector defining tasks for new job or trim restart
	= 0	to suppress:
		(1) flutter
		(2) flight dynamics
		(3) transient
TITLE(20)		title for job and case (80 characters)
CODE		alphanumeric code for job and case identification (4 characters)
OPUNIT	= 1	for English unit system (ft-slug-sec)
	= 2	for metric units system (m-kg-sec)
NROTOR		number of rotors (1 or 2)
WBTTAB	= 0	default (use empirical models)
	= 1	use wing-body-tail tables

DEBUG(25) integer vector controlling debug print:

values:
= 0 no debug print
= 1 trace print
= 2 low level print
= 3 high level print

elements:

(1) time (sec) at which debug print enabled
(2) input, 2-3 (INPTx)
(3) initialization, 2 (INITC, INITRx, INITB, INITE)
(4) trim iteration, 1-2 (TRIMI)
(5) loads, 2 (LOADIx)
(6) flutter matrices, 2-3 (FLUTM)
(7) flutter coefficients, 2-3 (FLUTIx, FLUTAx)
(8) flight dynamics, 2-3 (STABM, STABE)
(9) transient, 2 (TRANI)
(10) rotor/airframe motion and forces, 2-3 (RAMF)
(11) blade modes, 2 (MODE, MODEx)
(12) inertia coefficients, 2 (INRTCx)
(13) airframe constants and matrices, 2 (BODYC, ENGNC, MOTNCx, BODYMx, ENGNMx)
(14) induced velocity, 2 (WAKEUx, WAKENx)
(15) rotor matrices, 2-3 (INTRMx)
(16) hub/airframe motion and generalized forces, 2 (MOTNHx, BODYVx, ENGNVx, MOTNFx, MOTNS)
(17) rotor motion, 2-3 (MOTNRx)
(18) rotor aerodynamics, 2-3 (AEROFx, *AERBEDx*)
(19) blade section aerodynamics, 3 (AEROSx, *AEROSxB*)
(20) body forces and aerodynamics, 2 (BODYF)
(21) wake influence coefficients, 2 (WAKECx)
(22) vortex line and sheet, 3 (VTXL, VTXS,*VTXL2*)
(23) prescribed wake geometry, 2-3 (GEOMRx)
(24) free wake geometry, 1-3 (GEOMFx)
(25) timer, 1 (TIMER)

where the "x" can take the values 1 or 2 depending on which rotor is being examined.

VKTS		airspeed (knots), V
VEL		advance ratio, $V/(\Omega R)$

Note:

Input either VEL or VKTS by namelist input. If neither parameter is defined in the input namelist, $V = 0.$ is used.

VTIP		rotor 1 tip speed, ΩR (ft/sec or m/sec)
RPM		rotor 1 rotational speed (rpm)

Note:

Input either VTIP or RPM by input namelist. If neither parameter is defined, the normal tip speed, VTIPN, is used from namelist NLRTR. Rotor 2 speed is calculated from the gear ratio, TRATIO.

OPDENS		integer parameter defining specification of aerodynamic environment:
	$= 1$	given altitude and standard day
	$= 2$	given altitude and temperature
	$= 3$	given density and temperature
ALTMSL		altitude above mean sea level (ft or m) (for OPDENS = 1 or 2)
TEMP		air temperature (deg F or deg C) (for OPDENS = 2 or 3)
DENSE		air density (slug/ft^3 or kg/m^3) (for OPDENS = 3)
OPGRND		integer parameter controlling ground effect analysis
	$= 0$	out of ground effect
	$\neq 0$	in of ground effect
HAGL		altitude helicopter center of gravity above ground for ground effect analysis (ft or m)
OPENGN		integer parameter specifying engine state:
	$= 0$	for normal operation
	$= 1$	for autorotation (engine inertia, engine damping, and throttle control torque zero; no engine speed degree of freedom)
	$= 2$	for engine out (engine damping and throttle control torque zero)
AFLAP		wing flap angle, δ_F (deg)
RTURN		for free flight, trim turn rate, $\dot{\psi}_F$ (deg/sec) (positive to right)

initial values for controls (trimmed as appropriate)

COLL — collective stick displacement, δ_0 or $\Delta\theta_{govr}$ (deg), positive nose up

LATCYC — lateral cyclic stick displacement, δ_c (deg), positive left

LNGCYC — longitudinal cyclic stick displacement, δ_s (deg), positive aft

PEDAL — pedal displacement, δ_p (deg), positive right

APITCH — for free flight case: aircraft pitch angle, θ_{FT} (deg), positive nose up
for wind tunnel case: rotor shaft angle, θ_T (deg), positive nose up

AROLL — for free flight case: aircraft roll angle, ϕ_{FT} (deg), positive to right

(θ_{FT} and ϕ_{FT} define orientation of body axes relative to earth axes)

ACLIMB — for free flight case: aircraft climb angle, θ_{FP} (deg), positive up

AYAW — for free flight case: aircraft yaw angle, ψ_{FP} (deg), positive to right
for wind tunnel case: test module yaw angle, ψ_T (deg), positive to right

(θ_{FP} and ψ_{FP} define the orientation of the velocity axes relative to the earth axes: $V_{climb} = V\sin(\theta_{FP})$ and $V_{side} = V\sin(\psi_{FP})\cos(\theta_{FP})$)

MPSI — number of steps per revolution in motion and loads analysis (max 36). For nonuniform inflow, must be a multiple of the number of blades

MPSIR — number of azimuth steps between update of airframe vibration and rotor matrices in harmonic motion solution

MREV — number of revolutions between tests for motion convergence in harmonic motion solution

ITERM — maximum number of motion iterations

EPMOTN — tolerance for motion convergence (deg)

ITERC — maximum number of circulation iterations

EPCIRC — tolerance for circulation convergence ($\Delta C_T/\sigma$)

DOF(54) integer vector defining degrees of freedom used in vibratory motion solution.
values:
= 0 to omit
= 1 to use
order:

rotor 1	bending: (max 10) $q_1, q_2, ..., q_{10}$
rotor 1	torsion: (max 5) $p_0, p_1, ..., p_5$
rotor 1	gimbal : (max 1) β_g
rotor 2	bending: (max 10) $q_1, q_2, ..., q_{10}$
rotor 2	torsion: (max 5) $p_0, p_1, ..., p_5$
rotor 2	gimbal : (max 1) β_g
airframe	rigid : $\phi_F, \theta_F, \psi_F, x_F, y_F, z_F$
airframe	elastic: (max 10) $q_{s_7}, ..., q_{s_{16}}$
drive train	rotor/engine speed: ψ_s, ψ_i, ψ_e
drive train	governer: $\Delta\theta_t, \Delta\theta_{govr_1}, \Delta\theta_{govr_2}$

DOFT(8) integer vector defining blade bending degrees of freedom used for mean deflection (subset of DOF)
values:
= 0 to omit
= 1 to use
order:

rotor 1	bending: (max 4) q_1, q_2, q_3, q_4
rotor 2	bending: (max 4) q_1, q_2, q_3, q_4

MHARM(2) number of harmonics in rotor motion analysis (max 20)
(= 0 for mean only)
(1) rotor 1
(2) rotor 2

MHARF(2) number of harmonics in airframe vibration analysis (max 10)
(harmonics of N/rev)
(= 0 for static elastic only)
(suggest ≤ MHARM/NBLADE, and the same value for both rotors if coupled hub vibration used (see OPHVIB)
(1) rotor 1
(2) rotor 2

LEVEL(2) integer parameter specifying rotor wake analysis level
(must be consistant with INFLOW)

= 0	for unform inflow
= 1	for nonuniform inflow with prescribed wake geometry
= 2	for nonuniform inflow with free wake geometry

(1) rotor 1
(2) rotor 2

ITERU		number of uniform inflow wake-trim iterations at uniform inflow level.
	$= 0$	to skip
ITERR		number of nonuniform inflow with prescribed wake-trim iterations at nonuniform inflow/prescribed wake geometry level.
	$= 0$	to skip
ITERF		number of nonuniform inflow with free wake-trim iterations at nonuniform inflow/free wake geometry level.
NPRNTT		integer parameter controlling trim/performance/load print.
	≤ 0	suppress print (last iteration always printed)
	> 0	print every NPRNTT-th iteration
NPRNTP		integer parameter controlling performance print.
	≤ 0	suppress print
NPRNTL		integer parameter controlling loads print.
	≤ 0	suppress print
MTRIM		maximum number if iterations on controls to achieve trim
MTRIMD		number of iterations between update of trim derivative matrix
DELTA		control step in trim derivative matrix calculation (stick displacement, deg)
FACTOR		factor reducing control increment in order to improve trim convergence (typically .5)
EPTRIM		tolerance for trim convergence
OPGOVT		integer parameter specifying governor trim
	$= 0$	trim collective stick, δ_0
	$= 1$	trim rotor 1 governor
	$= 2$	trim rotor 2 governor
	$= 3$	trim both rotor governors

trim targets for wind tunnel cases:

CXTRIM	C_X/σ
XTRIM	X/q (ft^2 or m^2)
CTTRIM	C_T/σ or C_L/σ
CPTRIM	C_P/σ
CYTRIM	C_Y/σ
BCTRIM	β_c (deg)
BSTRIM	β_s (deg)

OPTRIM integer parameter specifying trim option

free flight cases:

$= 0$ no trim

$= 1$ trim forces and moments with:
$\delta_0, \delta_c, \delta_s, \delta_p, \theta_{FT}, \phi_{FT}$

$= 2$ trim forces and moments with:
$\delta_0, \delta_c, \delta_s, \delta_p, \theta_{FT}, \psi_{FP}$

$= 3$ trim forces, moments, and power with:
$\delta_0, \delta_c, \delta_s, \delta_p, \theta_{FT}, \phi_{FT}, \theta_{FP}$

$= 4$ trim forces, moments, and power with:
$\delta_0, \delta_c, \delta_s, \delta_p, \theta_{FT}, \psi_{FP}, \theta_{FP}$

$= 5$ trim symmetric forces and moments with:
$\delta_0, \delta_s, \theta_{FT}$

$= 6$ trim symmetric forces, moments, and power with:
$\delta_0, \delta_s, \theta_{FT}, \theta_{FP}$

wind tunnel cases:

$= 10$	no trim	
$= 11$	trim C_T/σ	with δ_0
$= 12$	trim C_T/σ	with θ_T
$= 13$	trim C_P/σ	with δ_0
$= 14$	trim β_c, β_s	with δ_c, δ_s
$= 15$	trim $C_T/\sigma, \beta_c, \beta_s$	with $\delta_0, \delta_c, \delta_s$
$= 16$	trim $C_L/\sigma, C_X/\sigma, C_Y/\sigma$	with $\delta_0, \delta_c, \delta_s$
$= 17$	trim $C_L/\sigma, C_X/\sigma, C_Y/\sigma$	with $\delta_0, \delta_c, \theta_T$
$= 18$	trim $C_L/\sigma, C_X/\sigma, \beta_c, \beta_s$	with $\delta_0, \delta_c, \delta_s, \theta_T$
$= 19$	trim $C_L/\sigma, X/q, C_Y/\sigma$	with $\delta_0, \delta_c, \delta_s$
$= 20$	trim $C_L/\sigma, X/q, C_Y/\sigma$	with $\delta_0, \delta_c, \theta_T$
$= 21$	trim $C_L/\sigma, X/q, \beta_c, \beta_s$	with $\delta_0, \delta_c, \delta_s, \theta_T$
$= 22$	trim β_c	with δ_s
$= 23$	trim $C_T/\sigma, \beta_c$	with δ_0, δ_s
$= 24$	trim $C_L/\sigma, C_X/\sigma$	with δ_0, δ_s
$= 25$	trim $C_L/\sigma, C_X/\sigma$	with δ_0, θ_T
$= 26$	trim $C_L/\sigma, C_X/\sigma, \beta_c$	with $\delta_0, \delta_s, \theta_T$
$= 27$	trim $C_L/\sigma, X/q$	with δ_0, δ_s
$= 28$	trim $C_L/\sigma, X/q$	with δ_0, θ_T
$= 29$	trim $C_L/\sigma, X/q, \beta_c$	with $\delta_0, \delta_s, \theta_T$

Additions to Namelist NLTRIM

FACTM	$0. \leq$ FACTM $\leq 1.$	lag factor to help convergence (default = 1.)
OPMXFWG	= 0	use original , $_{max}$ in free wake geometry calculations (default) (during rollup calculations, this may be superceded by ICORYCB variable)
	= 1	use positive , $_{max}$
	= 2	use negative , $_{max}$
	= 3	use outboard , $_{max}$
	= 4	use inboard , $_{max}$

If NLCFD is to be used, the following must be true:

OPREAD(2)=2

4.7 NLTRTR

Namelist NLTRTR

TITLE(20)	title for rotor and wake data (80 characters)
TYPE	rotor identification (4 characters) (suggest MAIN, FRNT, or RGHT for rotor 1; TAIL, REAR, or LEFT for rotor 2)
VTIPN	normal tip speed, ΩR_0 (ft/sec or m/sec)
RADIUS	blade radius, R (ft or m)
SIGMA	solidity, $\sigma = N * c_{mean}/(\pi R)$
GAMMA	Lock number, $\gamma_0 = \rho_0 a c_{mean} R^4/I_b$ (based on standard density, a = 5.7, and mean chord)

(γ and σ are only used to calculate the normalization parameters c_{mean} and I_b)

NBLADE	number of blades
TDAMP0	control system collective damping (ft-lb/rad/sec or m-N/rad/sec)
TDAMPC	control system cyclic damping (ft-lb/rad/sec or m-N/rad/sec)
TDAMPR	control system rotating damping (ft-lb/rad/sec or m-N/rad/sec)
NUGC	longitudinal gimbal natural frequency, ν_{GC}, or teeter natural frequency, ν_T (per rev at normal tip speed VTIPN)
NUGS	lateral gimbal natural frequency, ν_{GS} (per rev at normal tip speed VTIPN)
GDAMPC	longitudinal gimbal damping, C_{GC} or teeter damping C_T (ft-lb/rad/sec or m-N/rad/sec)
GDAMPS	lateral gimbal damping, C_{GS} (ft-lb/rad/sec or m-N/rad/sec)
LDAMPC	linear lag damper coefficient, C_ζ; estimated damping if a nonlinear damper is used (LDAMPM > 0.); the lag mode has structural damping also (GSB) (ft-lb/rad/sec or m-N/rad/sec)
LDAMPM	maximum moment of nonlinear lag damper, M_{LD}; (ft-lb or m-N)
= 0	to use linear lag damper

LDAMPR		lag velocity, $\dot{\zeta}_{LD}$, where maximum moment of lag damper occurs (rad/sec); hydraulic damping below $\dot{\zeta}_{LD}$ and friction damping above.
GSB(NBM)		bending mode structural damping, g_s
GST(NTM)		torsion mode structural damping, g_s

where NBM is the number of bending modes and NTM is the number of torsion modes as used in DOF.

ROTATE		integer parameter specifying rotor rotation direction as viewed from above.
	$= 1$	counter-clockwise.
	$= -1$	clockwise.
OPHVIB		integer parameter specifying hub vibration contributions; gravity and static velocity terms always retained.
		(= 0 to suppress)
		(1) vibration due to this rotor
		(2) vibration due to other rotor (must = 0 if $\Omega_2/\Omega_1 \neq 1$)
		(3) static elastic motion
BTIP		tip loss factor, B
OPTIP		integer parameter specifying tip loss type:
	$= 1$	tip loss factor
	$= 2$	Prandtl function
LINTW		integer parameter specifying twist type:
	$= 0$	nonlinear twist
	$\neq 0$	linear twist
TWISTL		linear twist rate, θ_{tw} (deg). used to calculate TWISTA and TWISTI if LINTW = 1
OPUSLD		integer parameter controlling use of unsteady lift, moment, and circulation terms:
	$= 0$	suppress terms
	$= 1$	include terms
	$= 2$	zero for stall (15 deg $< \|\alpha\| <$ 165 deg)
		(OPUSLD is set to zero internally if OPBED = 1)
OPCOMP		integer parameter controlling aerodynamic model:
	$= 0$	incompressible loads

Inflow Model

INFLOW(6) integer parameter defining induced velocity calculation:

(must be consistant with LEVEL)
(1) at this rotor: 0 for uniform, 1 for nonuniform

(2) at other rotor: 0 for zero, 1 for empirical,
2 for average at hub, 3 for nonuniform (only if $\Omega_2/\Omega_1 = 1$)

(3) at wing-body: 0 for zero, 1 for empirical,
2 for nonuniform

(4) at horizontal tail: 0 for zero, 1 for empirical,
2 for nonuniform

(5) at vertical tail: 0 for zero, 1 for empirical,
2 for nonuniform

(6) at point off rotor disk: 0 for zero, 1 for nonuniform

RROOT root vortex position for wake model, r_{root}/R

RGMAX $r_{\Gamma_{max}}$/R (induced velocity calculation using maximum bound circulation outboard of $r_{\Gamma_{max}}$/R).

Blade section aerodynamic characteristics

MRA number of radial segments (max 30)

RAE(MRA+1) radial stations r/R at edges of aerodynamic segments; sequential from root to tip.

The following quantities are specified at the midpoint of the aerodynamic segments:

CHORD(MRA) blade chord, c/R

XA(MRA) offset of aerodynamic center aft of the elastic axis, x_A/R. x_A is the point about which the moment data in the airfoil tables is given.

THETZL(MRA) incremental pitch of zero lift line, θ_{ZL} (deg); can be included in TWISTA; θ_{ZL} is the pitch of the axis corresponding to the zero angle of attack in the airfoil tables, relative to the twist angle (TWISTA)

TWISTA(MRA) blade twist relative .75R, θ_{tw} (deg)

XAC(MRA) offset of the aerodynamic center (for unsteady aerodynamics) aft of the elastic axis, x_{AC}/R

MCORRL(MRA)	Mach number correction factor $f_M = M_{eff}/M$ for lift
MCORRD(MRA)	Mach number correction factor $f_M = M_{eff}/M$ for drag
MCORRM(MRA)	Mach number correction factor $f_M = M_{eff}/M$ for moment
MRI	Blade section inertial and structural characteristics number of radial stations where characteristics defined (max 51)
RI(MRI)	radial stations r/R; sequential from root to tip, (RI(1) = 0. and RI(MRI) = 1.)
MASS(MRI)	section mass, m (slug/ft or kg/m)
EIXX(MRI)	chordwise bending stiffness (lb-ft^2 or N-m^2)
EIZZ(MRI)	flapwise bending stiffness (lb-ft^2 or N-m^2)
XI(MRI)	offset of center of gravity aft of elastic axis, x_I/R
XC(MRI)	offset of tension center aft of elastic axis, x_C/R (at the tip, XC should be set nearly equal to XI)
KP2(MRI)	polar radius of gyration about elastic axis, ${k_p}^2/R^2$
ITHETA(MRI)	section moment of inertia about elastic axis, I_θ (slug-ft or kg-m)
GJ(MRI)	torsional stiffness, GJ (lb-ft^2 or N-m^2)
TWISTI(MRI)	blade twist relative .75R, θ_{tw} (deg)

Stall Model

OPSTLL — integer parameter defining stall model:

- = 0 no stall
- = 1 static stall
- = 2 McCroskey stall
- = 3 McCroskey stall with dynamic stall vortex loads
- = 4 Boeing stall
- = 5 Boeing stall with dynamic stall vortex loads

(OPSTLL set = 0 internally if OPBED = 1)

(the stall delay can be suppressed by setting TAU = 0.)

OPYAW — integer parameter defining yaw flow corrections:

- = 0 both yawed flow and radial drag included
- = 1 no yawed flow ($\cos(\Lambda) = 0.$)
- = 2 no radial drag ($F_r = 0.$)
- = 3 neither yawed flow not radial drag included

TAU(3) — stall delay time constants for lift, drag, and moment: τ_L, τ_D, τ_M (calculated of < 0)

ADELAY — maximum angle of attack increment due to stall delay $\alpha_{maxdelay}$ (deg)

AMAXNS — angle of attack in linear range for no stall model α_{max} (deg)

PSID(3) — dynamic stall vortex load rise and fall time (azimuth increment) for lift, drag, and moment: $\Delta\psi_{ds}$ (deg)

ALFDS(3) — dynamic stall angle of attack for lift, drag, and moment: α_{ds} (deg)

ALFRE(3) — stall recovery angle of attack for lift, drag, and moment: α_{re} (deg)

CLDSP — maximum peak dynamic stall vortex induced lift coefficient: $\Delta C_{l_{ds}}$

CDDSP — maximum peak dynamic stall vortex induced drag coefficient: $\Delta C_{d_{ds}}$

CMDSP — maximum peak dynamic stall vortex induced moment coefficient: $\Delta C_{m_{ds}}$

KHLMDA — factor, κ_h, for hover induced velocity (typically 1.1)

KFLMDA — factor, κ_f, for forward flight induced velocity (typically 1.2)

FXLMDA — factor, f_x, for linear inflow variation in forward flight (typically 1.5)

FYLMDA — factor, f_y, for linear inflow variation in forward flight (typically 1.)

FMLMDA — factor, f_m, for linear inflow variation due to hub moment (typically 1.)

FACTWU — factor introducing lag in C_T, C_{M_x}, and C_{M_y} used to calculate induced velocity (typically .5)

KINTH — factor for hover interference velocity at other rotor (κ_{21} or κ_{12})

KINTF — factor for forward flight interference velocity at other rotor (κ_{21} or κ_{12})

(linear variation between KINTH at $\mu = .05$ and KINF at $\mu = .1$ is used)

KINTWB — factor for rotor-induced interference velocity at wing-body, K_W

KINTHT — factor for rotor-induced interference velocity at horizontal tail, K_H

KINTVT — factor for rotor-induced interference velocity at vertical, K_V

(K_W, K_H, and K_V equal fraction of fully-developed wake times maximum fraction surface in wake)

HINGE — integer parameter specifying blade mode type
- $= 0$ hinged
- $= 1$ cantilever
- $= 2$ articulated (flap and lag modes only)

NCOLB — number of collocation functions for bending mode calculations (total number of flap and lag, alternating); (max 20)

NCOLT — number of collocation functions for torsion mode calculations (max 10)

NONROT — integer parameter.
- $\neq 0$ to calculate nonrotating bending frequencies

EPMODE — criterion on change of collective pitch to update blade bending modes, $\Delta\theta_{75}$ (deg)

MASST — tip mass (slug or kg); the tip mass can also be directly in the section mass distrubution

XIT — offset of tip mass center of gravity aft of elastic axis, x_I/R

MBLADE — blade mass (slug or kg).
if ≤ 0 integral of section mass used (with mass included at r=0. to account for the hub mass)

EFLAP — flap hinge offset, e_f/R, (extent of rigid hub for cantilever blade)

ELAG — lag hinge offset, e_l/R, (extent of rigid hub for cantilever blade)

KFLAP — flap hinge spring (ft-lb/rad or m-N/rad)

KFLAG — lag hinge spring (ft-lb/rad or m-N/rad)

RCPLS — hinge spring parameter, $\Re_s$

TSPRNG — hinge spring parameter, θ_{so}

(hinge spring pitch angle is $\theta_s = \theta_{so} + \Re_s \theta_{75}$)

RCPL — structural coupling parameter, $\Re$.
(effective pitch angle, $\Re\theta$, used to calculate
blade bending modes (normally $\Re = 1.$))

NOPB — integer parameter specifying twist inboard of r_{FA}
(= 1 for no pitch bearing)

WTIN — integer parameter defining control system stiffness input:
= 1 for K_θ
= 2 for ω_θ

control system frequency, ω_θ
(per rev, at notmal tip speed, VTIPN):

FT0 — collective

FTC — cyclic

FTR — reactionless

control system stiffness, K_θ
(ft-lb/rad or m-N/rad):

KT0 — collective

KTC — cyclic

KTR — reactionless

KPIN — integer parameter defining pitch/bending coupling input:
= 1 for input
= 2 for calculated
(negative to suppress cosine terms factors in K_{P_i} and K_{P_G})

<u>root geometry to calculate pitch/bending coupling
(KPIN = 2 or -2)</u>

PHIPH — pitch horn cant angle, ϕ_{PH} (deg)

PHIPL — pitch link cant angle, ϕ_{PL} (deg)

RPB	pitch bearing radial location, r_{PB}/R
RPH	pitch horn radial location, r_{PH}/R
XPH	pitch horn length, x_{PH}/R
ATANKP(NBM)	pitch/bending coupling $\tan^{-1}(K_{P_i})$ (deg), for pitch horn level (KPIN = 1 or -1)
DEL3G	pitch/gimbal coupling $\tan^{-1}(K_{P_G})$ (deg), for pitch horn level.
RFA	feathering axis radial location, r_{FA}/R
ZFA	gimbal undersling, z_{FA}/R
XFA	torque offset, x_{FA}/R
CONE	precone angle, δ_{FA_1} (deg) (positive up)
DROOP	droop angle, δ_{FA_2} (deg), at $\theta_{75} = 0$, (positive down from precone)
SWEEP	sweep angle, δ_{FA_3} (deg), at $\theta_{75} = 0$, (positive aft)
FDROOP	feathering axis droop angle, δ_{FA_4} (deg), (positive down from precone)
FSWEEP	feathering axis sweep angle, δ_{FA_5} (deg), (positive aft)
MRM	number of radial stations for integration of inertial coefficients (max 50)
MRB	number of radial stations in blade mode calculation (max 50)

4.8 NLHHC

New Namelist NLHHC

$PSICOLL$	(deg)	ψ_c	$(default = 0.)$
$PSILAT$	(deg)	ψ_{lat}	$(default = 0.)$
$PSILON$	(deg)	ψ_{lon}	$(default = 0.)$
$THCOLL$	(deg)	θ_c	$(default = 0.)$
$THLAT$	(deg)	θ_{lat}	$(default = 0.)$
$THLON$	(deg)	θ_{lon}	$(default = 0.)$

Variables in this namelist are used only if $OPHHC = 1$ *in namlist NLHHC2.*

4.9 NLHHC2

New Namelist NLHHC2

OPHHC	= 0	*no HHC used (default)*
	= 1	*use HHC input from NLHHC*
	= 2	*use HHC input from NLHHC2*
NHHC		*(integer) number of HHC harmonics (max 12)*
HHCP0	*(deg)*	*(real) HHC "collective" (default = 0.)*
AHHC(NHHC)	*(deg)*	*(real) vector of HHC cosine amplitudes (default = 12*0.)*
BHHC(NHHC)	*(deg)*	*(real) vector of HHC sine amplitudes (default = 12*0.)*

4.10 NLHIRES

New Namelist NLHIRES

OPINT	$= 0$	*no HIRES performed (default)*
	$= 1$	*HIRES calculations performed*
ITERINT		*number of far wake iterations (default = 1)*
MPSIINT		*number of azimuth steps (max. 720)*
JFIRST		*index of first azimuth step (default = 1)*
JLAST		*index of last azimuth step*
MPSIWGP		*number of azimuths for wake geometry print*
FACTINT		*lag factor for ITERINT iterations (default = 1)*
DLSINT		*lifting surface correction (default = -1.)* *(see DLS in NLRTR for more detail.)*
MRAINT		*number of HIRES radial stations (max 100)*
RAEINT(MRAINT+1)		*edges of aerodynamic segments (re R)*
ITERNW		*number of HIRES near wake iterations* *(default = 0)*
FACTNW		*lag factor for ITERNW iterations (default = .5)*
KNWINT		*extent of HIRES near wake. Near wake extends* *from $\phi = 0$ to ϕ =KNWINT$*\Delta\psi$* *behind reference blade.*
OPNEGV		*negative tip vortex and inboard wake*
	$= 0$	*include the tip vortex and inboard wake in* *the "near wake".*
	$= 1$	*eliminate the tip vortex and inboard wake* *that is in the "near wake". (default)*
OPCSNW	$= 0$	*distributed core model in shed NW (default)*
	$= 1$	*concentrated core model in shed NW*

OPCTNW	$= 0$	*distributed core model in trailed NW (default)*
	$= 1$	*concentrated core model in trailed NW*
MDLSNW		*shed near wake model*
	$= 0$	*omit shed near wake*
	$= 1$	*line vortex w/ stepped circulation*
	$= 2$	*line vortex w/ linear circulation (default)*
MDLTNW		*trailed near wake model*
	$= 0$	*omit trailed near wake*
	$= 1$	*line vortex w/ stepped circulation*
	$= 2$	*line vortex w/ linear circulation (default)*
CORESNW		*near wake shed core size (re. R)*
	$< 0.$	*use default = .5*(panel length)*
		(default = .015)
CORETNW		*near wake trailed core size (re. R)*
	$< 0.$	*use default =.5*(panel width)*
		(default = .009)
OPNWMIN		*minimization of near wake panels option*
	$= 0$	*supress minimization (default)*
	$= 1$	*perform reduction in number of panels*
OPNWCRC		*updating of near wake*
	$= 0$	*update at each azimuth (default)*
	$= 1$	*update each revolution*
OPDCORE	$= 0$	*use single core model (default)*
	$= 1$	*use dual core model*
COREINT		*core size for single core HIRES model*
OPTVCOR		*single core expansion model*
	$= 0$	*constant size core (default)*
	$= 1$	*step fn. in far wake core size (HIRES only)*
	$= 2$	*core fn. in far wake core size (HIRES only)*
	$= 11$	*same as* $= 1$ *except also in trim loop*
	$= 12$	*same as* $= 2$ *except also in trim loop*
PHIINC		*wake age at which to increase core size to RCORINC by a step function*
RCORINC		*core size after age PHIINC*
RCOR0 to RCOR10		*coefficients of core size expansion function*

OPCORAC	= 0	*use constants RCORDCA and RCORDCB for inner and outer core sizes of dual core*
	= 1	*use DCORAC1 and DCORAC2 to calculate inner core size, use RCORDCB for outer core size in dual core*
RCORDCA		*inner core size (re. R)*
RCORDCB		*outer core size (re. R)*
DCORAC1		*used in calculating inner core size*
DCORAC2		*used in calculating inner core size*
NDCCODA		*number of terms in dual core expansion (inner)*
NDCCODB		*number of terms in dual core expansion (outer)*
DCCORFA0		*term in inner core expansion function*
DCCORFA(NDCCODA)		*term in inner core expansion function*
DCCORFB0		*term in outer core expansion function*
DCCORFB(NDCCODB)		*term in outer core expansion function*
GAMACST		*constant used in calculating inner core strength*
DRPROOT		*aero. coll. point offset at root (re R) (default = 0.)*
DRPTIP		*aero. coll. point offset at tip (re R) (default = 0.)*
DRP		*aero. coll. point offset on blade (re R) (default = 100 * 0.)*
OPSEGD		*segmentation of vortex elements*
	= 0	*off*
	= 1	*on (default)*
OPWFCOR		*read and use TF correction input files*
	= 0	*off (default)*
	= 1	*on*
WKMDL1(13)		*same as WKMODL(13) in NLRTR. This used for far iterations 1 thru ITERINT-1*
WMDLINT(13)		*same as WKMODL(13) in NLRTR. This used for last far wake iteration.*
NLINT		*(do not use)*
NGINT		*(do not use)*

MRGINT(NGINT) *(do not use)*

MRLINT(NLINT) *(no not use)*

4.11 NLBED

New Namelist NLBED

OPBED	$= 0$	*(integer) use Johnson aerodynamics (original CAMRAD)*
	$= 1$	*(integer) use Indicial aerodynamics*
HCOR		*(real) core size for trailed near wake (re. R)*
	$< 0.$	*use default core size* $(1/2)\Delta r$ *(default)*
	$\geq 0.$	*use this constant core size*
ICURV	$= 1$	*(integer) use circular arc trailed near wake (default)*
	$= 0$	*(integer) use straight trailed near wake*
ILEED	$= 1$	*(integer) "lead" terms in trailed wake calculation (default)*
	$= 0$	*(integer) do not use "lead" terms (not recommended)*

4.12 NLSWP

Namelist NLSWP

SWPLO(30)	*(deg)*	*(real) sweep of quarter chord line for use in low resolution aerodynamic corrections at aerodynamic collocation points (default = 30*0.)*

4.13 NLWAKE

Namelist NLWAKE

FACTWN		factor introducing lag in bound circulation used to calculate induced velocity.
OPVXVY		integer parameter: (= 0 to suppress x and y components of induced velocity calculated at the rotors)
KNW		extent of near wake, K_{NW}
KRW		extent of rolling up wake, K_{RW}
KFW		extent of far wake and tip vortices, K_{FW}
KDW		extent of far wake and tip vortices for points off rotor disk, K_{FW} (wake age ϕ =K$\Delta\psi$; all K $\geq$ 1)
RRU		initial radial station of wake rollup, r_{RU}/R
FRU		initial tip vortex fraction of , $_{max}$ for rollup, f_{RU}
PRU		extent of rollup in wake age, ϕ_{RU} (deg)
FNW		tip vortex fraction of , $_{M}$ for near wake, f_{NW}
DVS		sheet edge test parameter, d_{vs} ($\leq$ 0 to suppress test)
DLS		lifting surface correction parameter, d_{ls} ($\leq$ 0 to suppress correction)
CORE(5)		vortex core radii, r_c/R (1) tip vortices (2) burst tip vortices (3) tip vortices in far wake off rotor (4) trailed lines ($\leq$ 0 for default = s/2) (5) shed lines ($\leq$ 0 for default = t/2)
OPCORE(2)		integer parameter specifying vortex core type: values: = 0 for distributed vorticity = 1 for concentrated vorticity elements: (1) tip vortices (2) inboard wake

OPNWS(2) integer parameter controlling action when inflow and circulation points conincide in near wake ($\phi = 0$) and sheets are being used:
values:
$= 0$ to use two sheets
$= 1$ to use two lines
$= 2$ to use single sheet
elements:
(1) shed wake
(2) trailed wake

LHW number of spirals of far wake for axisymmetric case, L_{HW}

OPHW integer parameter: (= 0 for axisymmetric wake geometry)

OPRTS integer parameter: ($\neq 0$ to include rotation matrices (R_{TS},etc.) in influence coefficients)

WKMODL(13) integer parameter defining wake model:
values:
$= 0$ to omit element
$= 1$ for line segment with stepped circulation distribution
$= 2$ for line segment with linear circulation distribution
$= 3$ for vortex sheet element
elements:
(1) tip vortices (stepped line or linear line)
(2) near wake shed vorticity
(3) near wake trailed vorticity
(4) rolling up wake shed vorticity
(5) rolling up wake trailed vorticity
(6) far wake shed vorticity
(7) far wake trailed vorticity
(8) far wake (off rotor) shed vorticity
(9) far wake (off rotor) trailed vorticity
(10) bound vortices (no sheet model)
(11) axisymmetric wake axial vorticity (no line model)
(12) axisymmetric wake shed vorticity (no line model)
(13) axisymmetric wake ring vorticity (no line model)

MRG number of circulation points for near wake ($\leq$MRA)

NG(MRG) circulation points, identified by aerodynamic segment number: n_{G_i} for i = 1 to MRG (corresponding r_i must be between r_{root}/R and 1.)

MRL number of inflow points ($\leq$MRA)

NL(MRL) points at which the induced velocity is calculated, identified by aerodynamic segment number:
n_{L_i} for i = 1 to MRL

OPWKBP(3)		integer parameter controlling blade position for wake analysis: (1) = 0 to suppress inplane motion (2) = 0 to suppress all harmonics except mean (3) = 0 for linear from $r = r_{root}$ to $r = 1$.
VELB		core burst propogation rate, $V_b = \partial\phi/\partial\psi$
DPHIB		core burst age increment, $\Delta\phi_b$ (deg)
DBV		core burst test parameter, d_{bv} (< 0 to suppress bursting)
QDEBUG		velocity criterion for debug print: print if $\left\|\vec{V}\cdot\vec{k}/,\right\| >$ QDEBUG
		Prescribed Wake Geometry
KRWG		extent of prescribed wake geometry, K_{RWG} (age $\phi = K_{RWG}\Delta\psi$) (max 144)
OPRWG		integer parameter defining prescribed wake geometry model:
	= 1	from $K_1 = f_1\lambda, K_2 = f_2\lambda$, input K_3, input K_4
	= 2	option #1 without interference velocity in λ
	= 3	from K_1, K_2, K_3, K_4
		Langrebe prescribed wake geometry:
	= 4	from C_T
	= 5	from , $_{max}$
	= 6	from λ
	= 7	from λ without interference
		Kocurek and Tangler prescribed wake geometry:
	= 8	from C_T
	= 9	from , $_{max}$
	= 10	from λ
	= 11	from λ without interference
		factors f_1 and f_2 for prescribed wake geometry:
FWGT(2)		tip vortex
FWGSI(2)		inside sheet edge
FWGSO(2)		outside sheet edge
		constants K_1, K_2, K_3, K_4 for prescribed wake geometry:
KWGT(2)		tip vortex
KWGSI(2)		inside sheet edge
KWGSO(2)		outside sheet edge

Free Wake Geometry

KFWG		extent of free wake geometry distortion calculation, K_{FWG} (age $\phi = KFWG * \Delta\psi$); suggest (.4/μ)MPSI; *(maximum 144)*
OPFWG		integer parameter defining free wake geometry model:
	= 1	Scully free wake geometry
	= 2	same as = 1, without interference velocity
ITERWG		number of wake geometry iterations (suggest 2 or 3) *(suggest 4 or 5 for low speed cases)*
FACTWG		factor introducing lag in distortion calculation to improve convergence (suggest .5)
RTWG(2)		radial station r/R of trailed vorticity: (1) inside sheet edge (2) outside sheet edge, or trailed line (suggest .4)
WGMODL(2)		integer parameter defining wake model: values: = 0 to omit = 1 for line segment = 2 for sheet element elements: (1) inboard trailed wake elements (1) shed wake elements
COREWG(4)		vortex core radii, r_c/R (1) tip vortices (2) burst tip vortices ($\leq$ 0 for default = unburst value) (3) inboard trailed lines ($\leq$ 0 for default = .5*(RTWG(2) - RTWG(1))) (4) shed lines ($\leq$ 0 for default = 0.4$\Delta\psi$)
MRVBWG		number of wake revolutions used below point where induced velocity being calculated (suggest 2)
LDMWG		integer parameter, l_{DM}: general update every $l_{DM}\Delta\psi$ increment in boundary age (suggest 180 deg/$\Delta\psi$)
NDMWG(MPSI)		integer parameter, $n_{DM}(\psi_j)$: boundary update every n_{DM} increment in age, function of $\psi_j = j\Delta\psi$, j = 1 to MPSI; suggest 90 deg/$\Delta\psi$ for and aft; and 45 deg/$\Delta\psi$ on sides.

DQWG(2)	incremental velocity criteria: (suggest $0.04\lambda_1$ to $0.08\lambda_1$) (1) near wake elements defined by $\|\Delta\vec{q}\|$ >DQWG(1) (2) integrate bound vortex line in time over if $\|\Delta\vec{q}\|$ >DQWG(2)
IPWGDB(2)	integer parameters controlling debug level 3 print of wake geometry distortion (1) IPR: print distortion before general update every IPR*$\Delta\psi$; (= 0 to suppress) (1) IPRS: print distortion after each iteration every IPRS*$\Delta\psi$; (= 0 to suppress; last iteration printed in full)
QWGDB	parameter controlling debug level 3 print: induced velocity contribution of wake element printed if $\|\Delta\vec{q}\|$ > QWGDB; (suggest $0.5\lambda_1$ to $1.0\lambda_1$)

4.14 NLBURST

New Namelist NLBURST

OPBURST		*tip vortex bursting model*
	$= 0$	*no bursting calculations (default)*
	$= 1$	*bursting calculations performed*
PSITOL	$=$	*ψ tolerance used to define a "close" interaction (radians).*
	< 0	*to allow only BVI crossings to cause a bursting. (default)*
ZTOL		*height tolerance for bursting criteria (re R)*
CORMULT		*core size multiplication factor applied at a burst event.* $r_{c,burst} = r_{c,old} * CORMULT$
CIRMULT		*circulation multiplication factor applied at a burst event.* $, _{c,burst} = , _{c,old} * CIRMULT$ *(not fully functional yet)*

4.15 NLROLL

New Namelist NLROLL

OPROLLU	$= 0$	*no rollup calculation performed*
	$= 1$	*rollup calculations in low resolution only*
	$= 2$	*rollup calculations in high resolution only*
	$= 3$	*rollup calculations in both low and high resolution*
OPLOWR	$= 0$	*do not print low resolution rollup data*
	$= 1$	*write low resolution rollup data (circulation distribution, tip and secondary vortex positions, tip and secondary vortex radial circulation distributions)*
	$= 2$	*same as* $= 1$ *plus large core circulation data*
OPHIWR	$= 0$	*do not print hi resolution rollup data*
	$= 1$	*write hi resolution rollup data*
CORELG		*tip core size for the large core calculations (ref. radius) typically* $= 0.3$
ITERRUP		*number of iterations for convergence between large core calculations the wake geometry-trim iteration*
ITERFRU		*number of iterations between the wake geometry and the trim iteration when using the rollup results (corresponds to ITERF in the trim iteration without rollup).*
ITERLGC		*number of iterations between the induced velocity calculation and the large core circulation calculation (typically* $= 20$*)*
FLGCORG		*lag factor for the large core calculation iteration (typically* $= 0.1$*, similar to FACTWN in NLWAKE)*
NTCOR		*number of tip vortex cores for rollup calculation (max 10)*
TIPCORE(NTCOR)		*tip core sizes (ref. radius)*
NSCOR		*number of secondary vortex cores for rollup calculation (max 10)*
SECCORE(NTCOR)		*secondary core sizes (ref. radius)*
NTIPFCT		*number of coefficients in function to phase-in rollup calculation results. (max 10)*
TIPFC0		*coefficient in function to phase-in rollup results*
TIPFC(NTIPFCT)		*coefficients for function tip phase-in rollup results (for linear phase-in, TIPFC(1) = 1.0/(age in radians to complete phase-in process))*

ISPIN	= 0	*do not include spinning relation between tip and secondary vortex cores.*
	= 1	*include spinning relation between tip and secondary vortex cores.*
TAUC0		*coefficient to phase-in tip/secondary vortex spinning*
TAUC1		*coefficient to phase-in tip/secondary vortex spinning*
ISECPH	= 0	*no secondary phase-in (default)*
	= 1	*secondary phase-in (don't use)*
IRUZCOR		*z-correction in rollup calculations*
	= 0	*don't use (default; leave blank)*
	= 1	*use (do not use this, it's not functioning)*
NSEARCH		*(leave blank)*
IRUDZ		*(leave blank)*
OPROLSS		*phase-in model choice*
	= 0	*use phase-in function above (default)*
	= 1	*use Spreiter-Sachs model (do not use, it's not functioning)*
CROLSSXY		*(leave blank)*
CROLSSZ		*(leave blank)*
ICORYCB		*multi-core model choice*
	= 0	*use original multi-core model (default)*
	= 1	*single core variable, multi-core model*
	= 2	*multi core variable, multi-core model*
IFWLGC		*usage of large core circulation in free wake geometry*
	= 0	*use original circulation (default)*
	= 1	*use single weighted large core max circulation*
	= 2	*use double weighted large core max circulation*
COREXP		*core exponent for multi-core model (integer) (default = 1) (Scully-type core)*

4.16 NLCFD

New Namelist NLCFD

OPCFD	= 0	*do not perform the partial angle calculations*
	= 1	*compute partial angles (OPMOTN= 0) or partial inflow (OPMOTN = 1) and output information to the ALPHAP.DAT file*
	= 2	*same as = 1 except files CFDAERO.DAT and CAMAERO.DAT are read at start of run and used in the aerodynamic calculations.*
OPBVI	= 0	*if OPCFD= 1, do not remove tip vortex elements inside the BVI box from partial angle or inflow*
	= 1	*if OPCFD= 1, remove tip vortex elements inside the BVI box from partial angle or inflow, and append a table of tip vortex trajectory and strength to the file ALPHAP.DAT*
PHICFD	*(deg)*	*agewise extent of CFD box behind reference blade. used to limit vortex element testing for inclusin in the CFD box.*
RDB(6)		*CFD box dimensions relative to the blade tip (re. R) (all are positive numbers) (1) leading edge face (2) radial distance outboard of tip (outside edge face) (3) trailing edge face (4) radial station of inboard face (5) height of upper face (6) height of lower face*
BDB(6)		*BVI box dimensions relative to hub (re. R) (all are positive numbers and in shaft coordinates) (1) upstream face (2) starboard face (3) downstream face (4) port face (5) upper face (6) lower face*
OPMOTN	= 0	*if OPCFD= 1, output a partial angle table to the file ALPHAP.DAT, and if OPBVI= 1, transform the tip vortex segment endpoint coordinates to the "flapped blade" coordinate system before appending the wake table to ALPHAP.DAT*
	= 1	*if OPCFD= 1, output vertical velocity at the blade section to ALPHAP.DAT, and if OPBVI= 1, append a wake table with untransformed wake coordinates to ALPHAP.DAT*

4.17 NLMEAS

New Namelist NLMEAS

IMODEIN	$= 0$ $= 1$	*use regular blade motion calculation* *use input blade motion*
IAEROIN	$= 0$ $= 1$	*use regular aero calculation* *use input aero information*
FLAPFILE		*name of flap file (character*80)*
LAGFILE		*name of lag file (character*80)*
PITCHFILE		*name of pitch file (character*80)*
CNFILE		*name of cn file (character*80)*
NRADIN		*number of radials in FLAPFILE,LAGFILE,PITCHFILE* *(max = 50)*
RIN(NRADIN)		*radials in FLAPFILE,LAGFILE,PITCHFILE*
MPSIIN		*number of azimuths FLAPFILE,LAGFILE,PITCHFILE* *(max = 2049)*
NRADINA		*number of radials in CNFILE* *(max = 50)*
RINA(NRADINA)		*radials in CNFILE*
MPSIINA		*number of azimuths CNFILE* *(max = 2049)*
FLAP0		*flap angle for hinged case (deg)* *(default = 0.)*
LAG0		*lag angle for hinged case (deg)* *(default = 0.)*

4.18 NLBODY

Namelist NLBODY

TITLE(20)		title for airframe and drive train data (80 characters)
WEIGHT		aircraft gross weight including rotors (lb or kg)
		aircraft moments of inertia including rotors (slug-ft^2 or kg-m^2):
IXX		I_{xx}
IYY		I_{yy}
IZZ		I_{zz}
IXY		I_{xy}
IXZ		I_{xz}
IYZ		I_{yz}
TRATIO		ratio of rotor 2 rotational speed to rotor 1 rotational speed, Ω_2/Ω_1 (transmission gear ratio r_{I_1}/r_{I_2})
CONFIG		integer parameter specifying helicopter coniguration:
	= 0	for one rotor
	= 1	for single main rotor and tail rotor (rotor 2 is the tail rotor)
	= 2	for tandem main rotors (rotor 2 is the rear rotor)
	= 3	for tilting proprotor aircraft (rotor 2 is the left rotor)
ASHAFT(2)		shaft angle, θ_R (deg), positive readward: (1) rotor 1 (2) rotor 2
ACANT(2)		shaft cant angle, ϕ_R (deg), positive to right for main rotor; positive upward for tail rotor; positive inward in helicopter mode for tilt rotor: (1) rotor 1 (2) rotor 2
ATILT		nacelle tilt angle, α_P (deg) for tilting proprotor configuration only: (=0. for airplane mode, =90. for helicopter mode)
HMAST		rotor mast length from pivot to hub (ft or m), for tilting proprotor only
DPSI21		$\Delta\psi_{21}$ (deg); rotor 2 azimuth angle,ψ_2 when rotor 1 azimuth angle $\psi_1 = 0.$; must be 0. if $\Omega_2/\Omega_1 \neq 1.$
CANTHT		horizontal tail cant angle, ϕ_{HT} (deg), positive to left
CANTVT		vertical tail cant angle, ϕ_{VT} (deg), positive to right

		location (fuselage station, butt line, and waterline) of aircraft components relative to a body fixed reference system having an arbitrary orientation and origin; fuselage station (FS) positive aft, butt line (BL) positive to right, and waterline (WL) positive up (ft or m)
FSCG BLCG WLCG		aircraft center of gravity location
FSR1 BLR1 WLR1		rotor 1 hub location (right nacelle pivot location for tilting proprotor configuration)
FSR2 BLR2 WLR2		rotor 2 hub location
FSWB BLWB WLWB		wing-body center of action
FSHT BLHT WLHT		horizontal tail center of action
FSVT BLVT WLVT		vertical tail center of action
FSOFF BLOFF WLOFF		point off rotor disk (for induced velocity calculation
CONTRLZ(11)		control inputs (deg) for all sticks centered ($\vec{v_p} = 0.$)
	rotor 1	(1) θ_0
	rotor 1	(2) θ_{1c}
	rotor 1	(3) θ_{1s}
	rotor 2	(4) θ_0
	rotor 2	(5) θ_{1c}
	rotor 2	(6) θ_{1s}
	aircraft	(7) δ_f
	aircraft	(8) δ_e
	aircraft	(9) δ_a
	aircraft	(10) δ_r
	aircraft	(11) θ_t

description of control system (T_{CFE}); K parameters are gains (deg per stick deflection), $\Delta\psi$ parameters are swashplate azimuth lead angles (deg)

	<u>one rotor, single main rotor and tail rotor tilting proprotor configurations:</u>
K0CFE	K_0, collective stick to collective pitch.
KCCFE	K_c, lateral cyclic stick to cyclic or differential collective pitch.
KSCFE	K_s, longitudinal cyclic stick to cyclic pitch.
KPCFE	K_p, pedal to tail rotor collective or differential cyclic pitch.
PCCFE	$\Delta\psi_c$, lateral cyclic stick to cyclic pitch (one rotor, or main rotor and tail rotor configurations)
PSCFE	$\Delta\psi_s$, longitudinal cyclic stick to cyclic pitch
PPCFE	$\Delta\psi_p$, pedal to differential cyclic pitch (tilting proprotor configuration only)
	<u>tandem main rotor configuration</u>
KF0CFE	K_{F0}, collective stick to front collective pitch
KR0CFE	K_{R0}, collective stick to rear collective pitch
KFCCFE	K_{FC}, lateral cyclic stick to front cyclic pitch
KRCCFE	K_{RC}, lateral cyclic stick to rear cyclic pitch
KFSCFE	K_{FS}, longitudinal cyclic stick to front collective pitch
KRSCFE	K_{RS}, longitudinal cyclic stick to rear collective pitch
KFPCFE	K_{FP}, pedal to front cyclic pitch
KRPCFE	K_{RP}, pedal to rear cyclic pitch
PFCCFE	$\Delta\psi_{FC}$, lateral cyclic stick to front cyclic pitch
PRCCFE	$\Delta\psi_{RC}$, lateral cyclic stick to rear cyclic pitch
PFPCFE	$\Delta\psi_{FP}$, pedal to front cyclic pitch
PRPCFE	$\Delta\psi_{RP}$, pedal to rear cyclic pitch
	<u>aircraft controls (all configurations)</u>
KFCFE	K_f, collective stick to flaperon
KTCFE	K_t, collective stick to throttle
KACFE	K_a, lateral cyclic stick to ailerons
KECFE	K_e, longitudinal cyclic stick to elevator
KRCFE	K_r, pedal to rudder

NEM		number of airframe modes for which data supplied; (max 10).
QMASS(NEM)		generalized mass, M_k, including rotors (slug or kg)
QFREQ(NEM)		generalized frequency, ω_k (Hz)
QDAMP(NEM)		structural damping, g_s
QDAMPA(NEM)		aerodynamic damping, $F_{q_k \dot{q}_k} = \partial(Q_k/(\frac{1}{2}\rho V^2))/\partial(\dot{q}_{s_k}/V)$ (ft^2 or m^2)
QCNTRL(4,NEM)		control derivatives, $F_{q_k \delta} = \partial(Q_k/(\frac{1}{2}\rho V^2))/\partial\delta$ for $\delta_f, \delta_e, \delta_a, \delta_r$ (ft^2/rad or m^2/rad)
DOFSYM(NEM)		integer vector designating type of mode (only required for flutter analysis with OPSYMM $\neq 0$):
	> 0	for symmetric
	< 0	for antisymmetric
ZETAR1(3,NEM)		linear mode shape, $\vec{\xi}_k$ at rotor 1 hub (ft/ft or m/m)
ZETAR2(3,NEM)		linear mode shape, $\vec{\xi}_k$ at rotor 2 hub (ft/ft or m/m)
GAMAR1(3,NEM)		linear mode shape, $\vec{\gamma}_k$ at rotor 1 hub (rad/ft or rad/m)
GAMAR2(3,NEM)		linear mode shape, $\vec{\gamma}_k$ at rotor 2 hub (rad/ft or rad/m)
		pitch/mast-bending coupling (rad/ft or rad/m):
KPMC1(NEM)		$K_{MC_k} = -\partial\theta_{1c}/\partial q_{s_k}$ for rotor 1
KPMS1(NEM)		$K_{MS_k} = -\partial\theta_{1s}/\partial q_{s_k}$ for rotor 1
KPMC2(NEM)		$K_{MC_k} = -\partial\theta_{1c}/\partial q_{s_k}$ for rotor 2
KPMS2(NEM)		$K_{MS_k} = -\partial\theta_{1s}/\partial q_{s_k}$ for rotor 2

Aircraft aerodynamic characteristics:

	wing-body:	
LFTAW	L_α/q	$(ft^2/rad$ or $m^2/rad)$
LFTFW	L_{δ_f}/q	$(ft^2/rad$ or $m^2/rad)$
LFTDW	L_{δ_F}/q	$(ft^2/rad$ or $m^2/rad)$
AMAXW	α_{max}	(deg)
IWB	i_{WB}	(deg)
DRG0W	$f_{WB} = D_0/q$	$(ft^2$ or $m^2)$
DRGVW	f_{vert}	$(ft^2$ or $m^2)$
DRGIW	$\pi e l_w{}^2 = \partial(L/q)^2/\partial(D_i/q)$	$(ft^2$ or $m^2)$
DRGFW	$D_{0_{\delta_f}}/q$	$(ft^2/rad$ or $m^2/rad)$
DRGDW	$D_{0_{\delta_F}}/q$	$(ft^2/rad$ or $m^2/rad)$
MOM0W	M_0/q	$(ft^3$ or $m^3)$
MOMAW	M_α/q	$(ft^3/rad$ or $m^3/rad)$
MOMFW	M_{δ_f}/q	$(ft^3/rad$ or $m^3/rad)$
MOMDW	M_{δ_F}/q	$(ft^3/rad$ or $m^3/rad)$
SIDEB	Y_β/q	$(ft^2/rad$ or $m^2/rad)$
SIDEP	VY_p/q	$(ft^3/rad$ or $m^3/rad)$
SIDER	VY_r/q	$(ft^3/rad$ or $m^3/rad)$
ROLLB	N_{x_β}/q	$(ft^3/rad$ or $m^3/rad)$
ROLLP	VN_{x_p}/q	$(ft^4/rad$ or $m^4/rad)$
ROLLR	VN_{x_r}/q	$(ft^4/rad$ or $m^4/rad)$
ROLLA	$VN_{x_{\delta_a}}/q$	$(ft^3/rad$ or $m^3/rad)$
YAWB	N_{z_β}/q	$(ft^3/rad$ or $m^3/rad)$
YAWP	VN_{z_p}/q	$(ft^4/rad$ or $m^4/rad)$
YAWR	VN_{z_r}/q	$(ft^4/rad$ or $m^4/rad)$
YAWA	$VN_{z_{\delta_a}}/q$	$(ft^3/rad$ or $m^3/rad)$
	Horizontal tail:	
LFTAH	L_α/q	$(ft^2/rad$ or $m^2/rad)$
LFTEH	L_{δ_e}/q	$(ft^2/rad$ or $m^2/rad)$
AMAXH	α_{max}	(deg)
IHT	i_{HT}	(deg)
	Vertical tail:	
LFTAV	L_α/q	$(ft^2/rad$ or $m^2/rad)$
LFTRV	L_{δ_r}/q	$(ft^2/rad$ or $m^2/rad)$
AMAXV	α_{max}	(deg)
IVT	i_{VT}	(deg)

		Airframe interference:	
FETAIL		$f_\epsilon = \partial(L/q)/\partial\epsilon$	(ft^2 or m^2)
LHTAIL		horizontal tail length, l_{HT}, for ϵ (ft or m)	
HVTAIL		vertical tail height, h_{VT}, for σ, positive up (ft or m)	
OPTINT		integer parameter controlling airframe/tail aerodynamic interference:	
	$= 0$	to suppress ($\epsilon = 0.$ and $\sigma = 0.$)	
		Engine and Drive Train Parameters:	
ENGPOS		integer parameter specifying drive train configuration:	
	$= 0$	one rotor	
	$= 1$	asymmetric, engine by rotor 1	
	$= 2$	asymmetric, engine by rotor 2	
	$= 3$	symmetric	
IENG		engine rotational inertia, ${r_e}^2 I_e$, for both engines if symmetric configuration (slug-ft^2 or kg-m^2)	
		drive train spring constants (ft-lb/rad or m-N/rad)	
KMAST1		rotor 1 shaft, K_{M_1} or K_M	
KMAST2		rotor 2 shaft, K_{M_2}	
KICS		interconnect shaft, ${r_{I_2}}^2 K_I$ or ${r_I}^2 K_I$	
KENG		engine shaft, ${r_E}^2 K_E$	
GSE		engine shaft structural damping, g_s, (ψ_e degree of freedom)	
GSI		interconnect shaft structural damping, g_s, (ψ_I degree of freedom)	
KEDAMP		engine damping factor, κ: typically = 1.0 for turboshaft engines, or 10. for induction electric motors	
THRTLC		$\partial P_E/\partial\theta_t$ (dimensional), for both engines if symmetric configuration; if the throttle variable, θ_t is only used for the governor, just the products: $K_P \partial P_E/\partial\theta_t = -\partial P/\partial\dot{\psi}_s$ $K_I \partial P_E/\partial\theta_t = -\partial P/\partial\psi_s$ must be correct ($P = \Omega_R Q_R = \Omega_E Q_E$)	
		governor proportional feedback gains (sec):	
KPGOVE		to throttle, $K_P = -\partial\theta_t/\partial\dot{\psi}_s$	
KPGOV1		to rotor 1 collective, $K_P = -\partial\theta/\partial\dot{\psi}_s$	
KPGOV2		to rotor 2 collective, $K_P = -\partial\theta/\partial\dot{\psi}_s$	
		governor integral feedback gains:	
KIGOVE		to throttle, $K_I = -\partial\theta_t/\partial\psi_s$	
KIGOV1		to rotor 1 collective, $K_I = -\partial\theta/\partial\psi_s$	
KIGOV2		to rotor 2 collective, $K_I = -\partial\theta/\partial\psi_s$	

	governor time lag $\tau_1 = 2\zeta/\omega_n(sec)$:
T1GOVE	throttle
T1GOV1	rotor 1
T1GOV2	rotor 2

	governor time lag $\tau_2 = 1/{\omega_n}^2(sec^2)$:
T2GOVE	throttle
T2GOV1	rotor 1
T2GOV2	rotor 2

4.19 NLLOAD

Namelist NLLOAD

		Airframe vibration:
MVIB		number of stations for airframe vibration calculation and print. (max 10).
	≤ 0	to suppress
		airframe location for vibration calculation (ft or m):
FSVIB(MVIB)		fuselage station
BLVIB(MVIB)		butt line
WLVIB(MVIB)		water line
ZETAV(3,NEM,MVIB)		linear mode shape, $\vec{\xi}_k$, at airframe vibration stations (ft/ft or m/m)
MALOAD		integer parameter controlling print of motion and aerodynamics:
	$= 0$	to suppress
	< 0	for only plots
MHLOAD		integer parameter controlling print of hub and control loads:
	$= 0$	to suppress
MRLOAD		number of radial stations for blade section load calculation and print (max 20)
	≤ 0	to suppress
RLOAD(MRLOAD)		blade radial stations, r/R, for section loads
MHARML		number of harmonics in loads analysis (max 30):
	< 0	for no harmonic analysis; suggest about MPSI/3
NPOLAR		integer parameter n for polar plots: symbol printed every n-th step.
NWKGMP(4)		integer parameter controlling wake geometry printer plot:
	$= 0$	to suppress:
		(1) top view
		(2) side view
		(3) back view
		(4) axial convection
MWKGMP		number of azimuth stations at which wake geometry plotted (max 8):
	≤ 0	for no plots
JWKGMP(MWKGMP)		azimuth stations at which geometry plotted ($\psi = JWKGMP * \Delta\psi$)

NPLOT(75) integer parameter controlling printer-plots of motion and aerodynamics:

values:

= 0 for no plot
= 1 for time history
= 2 for polar plot
= 3 for both
(only time histories avaliable for 1-4 and 68-75)

elements:

(1) bending motion
(2) torsion motion
(3) maximum circulation
(4) λ off rotor
(5) α
(6) M
(7) Λ
(8) C_l
(9) C_d
(10) C_m
(11) $C_{d_{radial}}$
(12) ,
(13) U_p
(14) U_T
(15) U_R
(16) U
(17) θ
(18) ϕ
(19) lag
(20) flap
(21) α_{eff}, lift
(22) α_{eff}, drag
(23) α_{eff}, moment
(24) M_{eff}, lift
(25) M_{eff}, drag
(26) M_{eff}, moment
(27) λ_x
(28) λ_y
(29) λ_z
(30) interference λ_x
(31) interference λ_y
(32) interference λ_z
(33) u_G
(34) v_G
(35) w_G
(36) L/c
(37) D/c
(38) M/c
(39) D_r/c

(40) F_x/c
(41) F_r/c
(42) $F_z/c = C_T/\sigma$
(43) M_a/c
(44) $\tilde{F}_r/c$
(45) not used
(46) not used
(47) not used
(48) C_P/σ
(49) C_{P_i}/σ
(50) $C_{P_{int}}/\sigma$
(51) C_{P_o}/σ
(52)* L
(53)* D
(54)* M
(55)* D_r
(56)* F_x
(57)* F_r
(58)* $F_z = T$
(59)* M_a
(60)* $\tilde{F}_r$
(61) not used
(62) not used
(63) not used
(64)* P
(65)* P_i
(66)* P_{int}
(67)* P_o
(68) rotating frame root loads
(69) nonrotating frame hub loads
(70)* rotating frame root loads
(71)* nonrotating frame hub loads
(72) section loads, shaft axes
(73) section loads, principal axes
(74)* section loads, shaft axes
(75)* section loads, principal axes

*dimensional quantities

for polar plots, last digit of integer part of (value/increment) is printed, if it is a multiple of NPOLAR; the increment is defined as follows:

.01 plots 27-35
.10 plots 6, 8-16, 24-26, 36-51
1.0 plots 5, 7, 17-23, 52-61
10. plots 62-67

KFATIG		parameter, K, in fatigue damage calculation; suggest 3 or 4
SENDUR(18)		endurance limit, S_E, (dimensional force or moment)
CMAT(18)		material constant, C
EXMAT(18)		material exponent, M
		<u>rotating frame root loads:</u>
	(1)	inplane shear, f_x
	(2)	axial shear, f_r
	(3)	vertical shear, f_z
	(4)	flap moment, m_x
	(5)	lag moment, m_z
	(6)	control moment, m_c
		<u>nonrotating frame hub loads:</u>
	(7)	drag force, H
	(8)	side force, Y
	(9)	thrust, T
	(10)	roll moment, M_x
	(11)	pitch moment, M_y
	(12)	torque, Q
		<u>section loads (principal axes):</u>
	(13)	chord shear, f_x
	(14)	axial shear, f_r
	(15)	normal shear, f_z
	(16)	flapwise moment, m_x
	(17)	edgewise moment, m_z
	(18)	torsion moment, m_t

the S-N curve is approximated by $N = C/(S/S_E - 1)^2$
use $S_E < 0.$ or $C < 0.$ to suppress damage fraction calculation;
use $M < 0.$ to suppress equivalent peak-to-peak load calculation

		Far field rotational noise:
MNOISE		number of microphones (max 10):
	≤ 0	for no noise analysis
RANGE(MNOISE)		microphone range relative hub (ft or m)
ELVATN(MNOISE)		microphone elevation relative hub (deg), positive above rotor disk
AZMUTH(MNOISE)		microphone azimuth relative hub (deg), defined as for rotor azimuth
MHARMN(3)		number of harmonics: (1) in noise calculation (max 500) (2) in aerodynamic load harmonic analysis (suggest MPSI/3) (3) in print of noise (≤ 0 for no print)
MTIMEN(3)		number of time steps (≤ 0 to suppress): (1) in period of noise calculation (max 500) (2) increment in noise print (3) increment in noise plot
AXS(MRA)		blade cross section area, A_{xs}/c^2 at aerodynamic segments, for thickness calculation (typically .685 times thickness ratio)
OPNOIS(4)		integer parameter controlling noise calculation:
	$= 0$	to suppress
	$= 1$	for impulsive chordwise loading
	$= 2$	for distributed chordwise loading
		(1) lift noise (2) drag noise (3) radial force noise (4) thickness noise